

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Computer Network Security		
Lecturer (for lectures)		Milovanović I. Emina, Ćirić M. Vladimir		
Lecturer/associate (for exercises)		Milovanović I. Emina, Ćirić M. Vladimir		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The aim of the course is for the students to gain insight into the potential security weaknesses and obtain the basic knowledge to enhance the security, as well as to gain the knowledge and skills to apply security techniques in order to increase the security of computer networks			
Course outcomes	It is expected for the students to be able to analyze security aspects of computer networks, and to be able to plan the strategy to increase security. It is also expected that the student has a knowledge to use tools, as well as to implement security protocols in order to increase security of the network			
Course outline				
Theoretical teaching	Security objectives. Security mechanisms and services. Information theory and security. Perfectly secure ciphers. Cryptanalytic attacks. Classical cryptosystems. Historical overview of ciphers. Block ciphers. Structure of the Fiestel cipher. DES. Weaknesses of DES. 3DES. Basic concepts of number theory and finite fields. Galois fields of type GF(2 ⁿ). Modular polynomial arithmetic. AES. Cipher operation modes: ECB, CBC, CFB, OFB, CTR. Public key cryptography. RSA. Elliptic curve cryptography (ECC). Data integrity protection. Hash functions. Authentication techniques: HMAC, CBC-MAC, NMAC. Key generation and exchange protocols. Security protocols for data protection, integrity and authentication (Kerbos, SSL/TLS. IPsec, VPN, PK, PGP, S/MIME). Application of firewall tools in system protection. Attacks. Computer viruses. Security of wireless computer networks			
Practical teaching (exercises, OFE, study and research work)	Work on examples through a set of laboratory exercises. Network equipment protection. Implementation of cryptographic algorithms. Implementation of security protocols at the network, transport and application levels. Implementation of SSL/TLS and SSH protocols. RADIUS and AAA servers. Firewall. Configuration of virtual private networks (VPN). Protection of wireless networks. Typical attacks. Implementation of security protocols in OpenSSL. Secure email exchange.			
Textbooks/references				
	1	William Stallings, "Cryptography and Network Security: Principles and Practice (7th Edition)", Prentice hall, 2017		
	2	William Stallings – Network Security Essentials: Applications and Standards (6th Edition, 2023)		
	3	W. Richard Stevens, Kevin R. Fall – TCP/IP Illustrated, Volume 1: The Protocols (2nd Edition, 2011)		
	4	Michael Watkins, Kevin Wallace, "CCNA Security", Cisco Press, 2008, ISBN 9781587202209		
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, exercises, programming assignments			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	30
Exercises			Oral exam	30
Colloquia				
Projects		40		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Secure Software Design		
Lecturer (for lectures)		Janković S. Dragan, Rajković J. Petar		
Lecturer/associate (for exercises)		Rajković J. Petar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The main course objective is introducing students with basic concepts and applied techniques used for secure code development			
Course outcomes	The students have to be able to identify the main problems related to software solutions security and to apply learned resolution techniques.			
Course outline				
Theoretical teaching	Introduction. Passwords. Pseudorandom numbers. Pseudorandom number generators. Buffer Overflow problem and ways to overcome it. Safety increase techniques. Shared resources deadlock problems. Input validation. SQL injection. Cryptography elements. Authentication protocols. Software configurability. Processing sensitive data. Memory management (allocation, deallocation). Viruses and malicious software.			
Practical teaching (exercises, OFE, study and research work)	Weak points in shell scripts. Password coding. Pseudorandom numbers based applications. Implementation of random generators. Methods for the Buffer Overflow problem overcoming. Access control matrix implementation. Thread management. Methods against file level deadlocks. Input validation. Authentication protocols . Using cryptographic algorithms in software. Configurations - design and using. Problems with memory allocation and de-allocation.			
Textbooks/references				
	1	Loren Kohnfeler, Designing Secure Software: A Guide for Developers, No Starch Press, 2021.		
	2	Peter Gutmann: Cryptographic Security Architecture, Design and Implementation, ISBN 038795387-6, Springer, 2003.		
	3	Michael Howard, David LeBlanc: Writing Secure Code, ISBN 073561722-8, Microsoft Press, 2002.		
	4	Erik Fretheim, Marie Deschene, Secure Software Systems, Jones&Bartlett Learning, 2023.		
	5	John Viega, Gary McGraw: Building Secure Software, ISBN 020172152-X, Addison-Wesley, 2001.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, Auditive exercises, Laboratory exercises			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	30
Exercises			Oral exam	20
Colloquia				
Projects		50		

Specification for the book of courses				
Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Digital Forensics		
Lecturer (for lectures)		Rančić D. Dejan, Predić B. Bratislav		
Lecturer/associate (for exercises)		Predić B. Bratislav		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Introduction to the process of identifying, preserving, and analyzing digital evidence, as well as preparing it for presentation in court in a forensically sound manner.		
Course outcomes		Theoretical and practical knowledge of the process of identifying, preserving, and analyzing digital evidence. Familiarity with software and hardware tools used in digital forensics. Understanding of the legal aspects of digital forensics. Knowledge of the fundamental principles, rules, and methodologies of digital forensics.		
Course outline				
Theoretical teaching		Introduction to Digital Forensics – Fundamentals and importance in the modern ICT environment. Legal Framework and Ethics – Legality of collecting and using digital evidence. Forensic Methodology – Standards and procedures for examining digital systems. Evidence Identification and Preservation – Methods for protecting data integrity. Digital Evidence Analysis – Techniques for interpreting and correlating data. Reporting and Presentation – Forensically sound preparation of evidence. Data Recovery – Restoring deleted or damaged files. Decryption – Techniques for breaking encryption and accessing data. OS and File System Forensics – Analyzing traces in system structures. Network Forensics – Monitoring and analyzing network traffic. Mobile Forensics – Examination of smartphones and tablets. Tools and Environments – Hands-on work with EnCase, FTK, VMware, and other tools.		
Practical teaching (exercises, OFE, study and research work)		Protection and Cloning of Digital Evidence – Practical application of tools for preserving the original state of media. Forensic File System Analysis – Examination of NTFS and EXT structures to uncover traces of activity. Deleted Data Recovery – Use of specialized software to restore deleted files. System Log and Timeline Analysis – Reconstruction of events based on log files. Network Traffic Examination – Use of tools like Wireshark to monitor and analyze network communication. Mobile Device Forensics – Extraction and analysis of data from Android and iOS devices. Password Cracking and Decryption – Application of brute-force and dictionary attacks on protected data. Creation and Verification of Forensic Report – Writing reports based on analyzed evidence. Use of EnCase and FTK Tools – Basic operations within leading forensic software platforms. Court Testimony Simulation – Presentation of evidence and expert witnessing in a simulated legal proceeding.		
Textbooks/references				
		1 Digital Forensics, Investigation, and Response (4th Edition), Chuck Easttom, ISBN: 978-1284226065		
		2 Advances in Digital Forensics XX, Elizabeth Kurkowski and Sujeet Shenoj, ISBN 978-3031710247		
		3 Cybercrime and Digital Forensics: An Introduction (3rd Edition), Thomas J. Holt, Adam M. Bossler, Kathryn C. Seigfried-Spellar, ISBN 978-0367360078		
		4		
		5		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures, auditory exercises, and student projects		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		10	Written exam	
Exercises		50	Oral exam	40
Colloquia				
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Cloud Computing		
Lecturer (for lectures)		Stojanović M. Natalija, Ćirić M. Vladimir		
Lecturer/associate (for exercises)		Stojanović M. Natalija		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The aim of the course is for students to gain insight into virtualization concepts and cloud system architecture, as well as service-oriented principles of system design and application development in cloud computing environments.			
Course outcomes	Upon successful completion of the course, the student will be able to explain the fundamental principles of virtualization, cloud computing, and high-performance computing. The student will be able to analyze requirements and design basic cloud infrastructure, apply virtualization, containerization, and orchestration technologies, and use IaaS, PaaS, and FaaS services. The student will be able to apply MapReduce, Hadoop, YARN, and Spark for distributed processing and analysis of big data, as well as technologies for distributed data storage. The student will be able to deploy, configure, and monitor the execution of distributed applications on clusters and in the cloud, and analyze and improve their performance and scalability.			
Course outline				
Theoretical teaching	Hardware virtualization. Cloud layers and types. Components and design of cloud infrastructure. Cloud infrastructure management. IaaS and PaaS providers. Migration. Amazon Web Services (AWS). Application containerization. Docker. Orchestration. Kubernetes. High-performance computing on computer clusters and in the cloud – MapReduce concept and programming model. Methods and technologies for storing big data on clusters and in the cloud (GFS, HDFS, etc.). Systems for big data processing in the cloud – Hadoop platform. Application execution in the cloud using the YARN resource management system. Systems for processing and analyzing large volumes and streams of data – Spark platform. Deployment, configuration, execution monitoring, and performance optimization of distributed software systems on clusters and in the cloud. Function as a Service (FaaS) in the cloud (AWS Lambda, Azure Functions, OpenWhisk, OpenFaaS, etc.).			
Practical teaching (exercises, OFE, study and research work)	Virtualization and use of virtual hardware. VirtualBox, VMware. Management of virtual machines in a private cloud. VMware ESXi, vSphere. Management of virtual machines in a public cloud. Amazon AWS. Application containerization in Docker. Docker files. Container management. Kubernetes. Implementation of distributed software systems using the MapReduce programming model. Storage of big data on computer clusters and in the cloud using HDFS. Development of Hadoop MapReduce applications; MapReduce data types and formats (input and output). Implementation and management of Hadoop application execution on YARN clusters. Implementation and management of Spark application execution on computer clusters. Performance improvement and evaluation of highly distributed software systems on clusters and in the cloud. Implementation of highly distributed software systems using Functions as a Service (OpenFaaS, Azure Functions).			
Textbooks/references				
	1	Anthony T. Velte, Toby J. Velte, Robert Elsenpeter, "Cloud Computing: A Practical Approach", Mc. Graw Hill, 2010, ISBN: 978-0-07-162695-8		
	2	D. C. Marinescu, Cloud Computing: Theory and Practice, 3rd Edition, Morgan Kaufmann, 2022.		
	3	Brendan Burns, Joe Beda, Kelsey Hightower, Lachlan Evenson, "Kubernetes: Up and Running: Dive into the Future of Infrastructure", O'Reilly Media; 3rd edition, 2022.		
	4	Tom White, Hadoop: The Definitive Guide, 4th Edition, O'Reilly Media, 2015.		
	5	Y. Demchenko, J. J. Cuadrado-Gallego, O. Chertov, M. Aleksandrova, Big Data Infrastructure Technologies for Data Analytics: Scaling Data Science Applications for Continuous Growth, Springer, Cham, 2024.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0

Teaching methods	Lectures, exercises, independent student work on seminar papers and projects		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	40
Exercises	40	Oral exam	
Colloquia			
Projects	20		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Virtualization		
Lecturer (for lectures)		Tošić B. Milorad		
Lecturer/associate (for exercises)		Tošić B. Milorad		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Operating systems			
Course objectives	Introducing students to the fundamentals of virtualization, types of virtualization, and technologies for implementing virtualization.			
Course outcomes	Students are able to design, configure, and use virtualized computing environments, including virtual machines, networks, data storage, and containers. They are trained to apply Docker and Kubernetes technologies, manage virtualized resources, and develop and deploy microservice applications in virtualized and cloud infrastructure.			
Course outline				
Theoretical teaching	What is virtualization: history, conceptualization, and basic principles. Types and technologies of virtualization. Operating system-level virtualization. Containers. Server virtualization. Storage virtualization. Network virtualization. Application virtualization. Virtual network functions. Virtual machines and hypervisors. Resource management systems in virtualized environments. Architectures of virtualized systems. Virtualization in IoT and devices with limited processing power.			
Practical teaching (exercises, OFE, study and research work)	Comparative analysis of existing open-source virtualization software. Installation of virtualization software. Creating a virtual machine. Creating a virtual network. Connecting two virtual machines in a LAN. Routing the internal virtual network and providing internet access. Using Docker for server virtualization. Kubernetes. Infrastructure as a service. Infrastructure for continuous development and deployment of software solutions. Implementation of an illustrative microservices-based application on a virtualized infrastructure. Commonly used architectural patterns. Network applications in next-generation architectures (6G+).			
Textbooks/references				
	1	Khedher, Omar. Mastering OpenStack: Implement the Latest Techniques for Designing and Deploying an Operational, Production-Ready Private Cloud. 3rd ed. Packt Publishing, 2024.		
	2	Zhang, Ying. Network Function Virtualization: Concepts and Applicability in 5G Networks. John Wiley & Sons, 2017.		
	3	Kavis, Michael J. Accelerating Cloud Adoption: Optimizing the Enterprise Speed and Agility. O'Reilly Media, 2020.		
	4	Bartlett, Jonathan. Cloud Native Applications with Docker and Kubernetes: Design and Build Cloud Architecture and Applications with Microservices, EMQ, and Multi-Site Configurations. Apress, 2023.		
	5	Portnoy, Matthew. Virtualization Essentials. 3rd ed. John Wiley & Sons, 2023.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, exercises, programming assignments			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures	30	Written exam		
Exercises	30	Oral exam		40
Colloquia				
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		System Administration		
Lecturer (for lectures)		Tošić B. Milorad		
Lecturer/associate (for exercises)		Tošić B. Milorad		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Operating systems, Computer networks			
Course objectives	Introducing students to basic procedures in the administration of systems based on Linux operating system principles.			
Course outcomes	Upon completion of the course, the student should be qualified to administer systems running the Linux operating system at a level sufficient to support the operational needs of medium-sized development teams.			
Course outline				
Theoretical teaching	Linux operating system. Basic principles of the Linux operating system from a system administration perspective. Overview of available open-source Linux distributions. Command-line interface and scripting languages. User management. File system operations. Operating system exposure. Server management – configuration files. Activity monitoring – logs. Principles of technical support: basic technical support principles. Remote support. Administration of virtualized systems: main types of virtualization and available technologies. Container-based systems. Virtual machine-based systems. Management of virtualized networks.			
Practical teaching (exercises, OFE, study and research work)	The course is conducted through three smaller practical projects completed individually by each student. Student presentations on the results of their individual projects. A report on the usage of artificial intelligence services and their usefulness for problems in the field of system administration.			
Textbooks/references				
	1	Nemeth, Evi, Garth Snyder, Trent R. Hein, Ben Whaley, and Dan Mackin. UNIX and Linux System Administration Handbook. 5th ed. Addison-Wesley Professional, 2022.		
	2	Ondara, Neville Asiago. Beginning Linux SysAdmin: Getting Started with Linux System Administration. Apress, 2026.		
	3	Both, David. Using and Administering Linux: Volume 2: Zero to SysAdmin: Advanced Topics. 2nd ed. Apress, 2023.		
	4			
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, exercises, programming assignments			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures	30	Written exam		
Exercises	30	Oral exam		40
Colloquia				
Projects				

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering	
Type and level of studies			Master Academic Studies	
The name of the course			Cryptography	
Lecturer (for lectures)			Rančić D. Dejan, Vučković V. Vladan	
Lecturer/associate (for exercises)			Dimitrijević M. Aleksandar	
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Introducing students to the field of cryptography and familiarizing them with the basic principles, algorithms, and standards used in this field.			
Course outcomes	Theoretical and practical knowledge of the basic principles, algorithms and standards used in the field of cryptography.			
Course outline				
Theoretical teaching	Introduction to cryptography and historical overview of the field. Mathematical foundations. Encryption using symmetric keys. Stream ciphers. Block ciphers. DES – The Data Encryption Standard. AES – The Advanced Encryption Standard. Encryption using a public-private key pair. Hash functions and data integrity. Digital signature schemes. RSA signing. The hash-and-sign paradigm. Certificates and public key infrastructures. Secure Socket Layer (SSL) and Transport Layer Security (TLS) standards.			
Practical teaching (exercises, OFE, study and research work)	Practical work on programming cryptographic elements using the OpenSSL library.			
Textbooks/references				
1	J. Katz and Y. Lindell, Introduction to Modern Cryptography, 3rd ed. New York, NY, USA, 2020, Chapman and Hall/CRC			
2	A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.			
3	Mark Stamp, “Information Security Principles and Practice“, John Wiley and Sons, Inc, New Jersey, U.S.A., 2006.			
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, auditory exercises, independent work of students on projects.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	30
Exercises			Oral exam	30
Colloquia		20		
Projects		20		

Specification for the book of courses

Study program	Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering
Type and level of studies	Master Academic Studies
The name of the course	High-performance Computing
Lecturer (for lectures)	Stojanović M. Natalija, Milentijević Z. Ivan
Lecturer/associate (for exercises)	Stojanović M. Natalija
Lecturer/associate (for OFE)	
Number of ECTS	6 Course status (obligatory/elective) Obligatory / Elective
Prerequisites	Parallel and distributed systems
Course objectives	Understanding the concepts and technologies of high-performance computing, as well as acquiring both theoretical and practical knowledge necessary for the development and analysis of high-performance applications on modern computing architectures.
Course outcomes	Upon successful completion of the course, students will be able to explain the fundamental concepts, methods, and techniques of High-Performance Computing (HPC) and analyze different HPC architectures and environments. Students will be able to apply programming models and technologies for parallel programming on multicore, GPU, multi-GPU, and hybrid architectures, as well as in distributed, cloud, and Edge computing environments. Students will be able to develop and configure HPC applications using appropriate technologies, such as OpenMP, CUDA, ROCm, OpenACC, MPI, Hadoop, and Spark. Students will be able to apply HPC technologies in artificial intelligence applications and other application domains. Students will be able to measure, analyze, and evaluate the performance of HPC applications, including execution time and energy consumption, and apply appropriate techniques to improve their performance, scalability, and efficiency. Students will be able to explain the fundamental principles of quantum computing and its potential applications.
Course outline	
Theoretical teaching	Overview of advanced concepts, methods, and techniques in high performance computing. High performance computing on multicore computers with shared memory (OpenMP, OpenCL, Intel TBB). Performance optimization on many-core computing architectures such as Graphics Processing Units (GPUs) (CUDA, OpenACC, Thrust, ROCm, etc.). High performance computing in distributed environments: clusters, grids, and cloud computing (MPI, Hadoop, Spark, etc.). Achieving high performance on hybrid computing architectures. High performance computing on multi-GPU architectures. Achieving high performance on edge computing systems. Applications of high performance computing in artificial intelligence (CUDA-X, cuDNN library, etc.). Quantum computing – principles, technologies, and applications. Analysis, evaluation, and optimization of HPC system performance (execution time, energy consumption), as well as programmer effort, using appropriate technologies. Applications of HPC in image and video processing, geospatial data analysis, environmental protection, business systems, bioinformatics, etc.
Practical teaching (exercises, OFE, study and research work)	Methods and technologies for configuring and monitoring application execution, as well as evaluating and analyzing performance. Implementation of high-performance applications on multicore computers with shared memory using OpenMP and OpenCL technologies. Implementation of high-performance applications on GPU architectures using CUDA and ROCm technologies. Implementation of high-performance applications on GPU architectures using OpenACC and Thrust technologies. Performance analysis of applications on computer clusters using MPI technology. Performance analysis of applications on clusters using Hadoop and Apache Spark technologies. Performance optimization of high-performance software systems on hybrid architectures using MPI + OpenMP. Achieving high application performance on multi-GPU architectures. Achieving high application performance application on edge computing systems, such as the NVIDIA Jetson Nano. Examples of applying high performance computing in artificial intelligence applications using the cuDNN library. Examples of applying HPC in AI applications using the Spark MLlib library. Implementation of an application in a selected domain, followed by performance optimization, analysis, and evaluation.
Textbooks/references	
1	W.M. Hwu, D. Kirk and I. E. Hajj, Programming Massively Parallel Processors: A Hands-on Approach, Elsevier, fourth edition, 2023.
2	G. Barlas, Multicore and GPU Programming An Integrated Approach, Second Edition, 2023
3	P Czarnul, Parallel Programming for Modern High Performance Computing Systems, 2018.

4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, exercises, independent student work on seminar papers and projects			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		50	Oral exam	30
Colloquia				
Projects		20		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Database Management Systems		
Lecturer (for lectures)		Stoimenov V. Leonid, Stanimirović S. Aleksandar		
Lecturer/associate (for exercises)		Stanimirović S. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Mastering knowledge of the basic concepts and principles of database management systems (DBMS) and their components. Gaining knowledge of advanced DBMS techniques (triggers, security, query optimization) and DBMS functions (replication, recovery, backups, fragmentation). Acquiring understanding of the fundamental concepts and principles of operation of advanced database systems, systems for handling large volumes of data, and large distributed systems			
Course outcomes	Theoretical and practical knowledge of database management systems, their components, and methods of use. Theoretical and practical knowledge of advanced DBMS techniques, administration, performance tuning, query optimization, recovery, backups, and data maintenance. Theoretical and practical knowledge of systems for handling large volumes of data and distributed systems supporting social networks and other large-scale systems.			
Course outline				
Theoretical teaching	Introduction to database management systems (DBMS). The role and responsibilities of the database administrator. Database management system: overview of architecture, core modules and functions, examples of such systems. Stored procedures and triggers. Query processing and optimization: concept of query optimization, relational algebra and execution plans, static and dynamic optimization, system catalog, database statistics and optimization. Indexes and index structures, multidimensional indexes, the role of indexes, and recommendations for proper use. Database system security: concept of database system security, security in DBMS, user privileges – granting and revoking (GRANT and REVOKE commands), privilege propagation, view-level security, statistical databases, DAC and MAC security mechanisms. Recovery and transactions. Replication. Backups. Distributed systems and systems for handling large volumes of data. Overview of features in open-source and commercial systems. Overview of DBMS functionalities in object-oriented databases, object-relational DBMS, and NoSQL databases.			
Practical teaching (exercises, OFE, study and research work)	The role of the database administrator, practical use of DBMS and administrative tools – practical examples and tasks. Database performance tuning, creating indexes. Query optimization – practical examples, issues, use of DBMS tools. Security in DBMS and the role of the database administrator, security at the operating system, network, hardware level, etc. – practical examples and tasks. Modern challenges for DBMS – example use of new technologies (object-relational mappers and their use: Hibernate/NHibernate, XML, etc.). NoSQL databases (concept, basic principles, classification, examples) and differences compared to traditional DBMS. Use of systems for big data processing and distributed systems.			
Textbooks/references				
	1	R.Ramakrishnan, J. Gehrke, Database Management Systems,Third Edition, 2002, ISBN-13: 978-0072465631, McGraw-Hill.		
	2	R. Elmasri, S. Navathe, Fundamentals of Database Systems, Pearson; 7 edition (2016), ISBN-13: 9780133970777		
	3	T. M. Conolly, C. E. Begg, Database Systems: A Practical Approach to Design, Implementation and Management, пето издање, 2009		
	4	G. Harrison, Next Generation Databases: NoSQLand Big Data, 1st edition, Apress, 2015, ISBN-13: 978-1484213308		
	5	Literature from the Internet on modern trends in database management systems.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, whiteboard exercises, lab exercises, independent student work on homework assignments and projects, student seminars (presentations of student work followed by discussions).			

Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises	70	Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Management of IT Resources and Services		
Lecturer (for lectures)		Milentijević Z. Ivan		
Lecturer/associate (for exercises)		Milentijević Z. Ivan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	To enable students to acquire knowledge about the role of computer resources and services in organizations, and about ensuring high-quality and reliable IT services.			
Course outcomes	Upon completing the course, students will be able to organize computer resources and services within a specific enterprise/corporation organizational framework, as well as plan the delivery of IT services.			
Course outline				
Theoretical teaching	Management of computer resources; administration methods and systems. Criteria and requirements for computer resources and services within a corporation. Capacity planning: technical infrastructure, service delivery, and budgeting. Capacity and plan control: methods and tools for monitoring technical capacity aspects and performance of networks, servers, workstations, software licenses, information systems, and human resources. Information security: operational procedures, standards, and tools for ensuring data protection. Operational and IT risks: compliance with regulatory requirements. IT control and auditing: service management standards (ITIL, COBIT, ISO, PSP/TSP). Lifecycle management of IT services: service operations, and planning/administration in multi-service environments.			
Practical teaching (exercises, OFE, study and research work)	Collection, organization, and analysis of requirements for computer resources and services. Capacity planning of computer resources: service planning, IT budgeting (capital expenditures, operational costs, and depreciation). Performance testing and control of computer resources: metrics, tools, and methodologies. Implementation examples and characteristics of standard operating procedures; information security testing tools. IT audit execution process: external and internal audits, examples of standards.			
Textbooks/references				
	1	Sutherland, Keith D., and J. Lawrence. A Practical Guide to Service Management: Insights from industry experts for uncovering, implementing, and improving service management practices. Packt Publishing Ltd, 2023.		
	2	Smith, Kevin J. The practical guide to world-class IT service management. Outskirts Press, 2017.		
	3	Maleh, Yassine, and Abdelkebir Sahid, eds. Navigating IT Governance for Resilient Organizations. IGI Global, 2024.		
	4			
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, practical sessions, consultations, and practical project tasks completed individually or in small teams. Presentation and discussion of project solutions.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		10	Written exam	
Exercises		50	Oral exam	40
Colloquia				
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Blockchain Technologies		
Lecturer (for lectures)		Janković S. Dragan, Ćirić M. Vladimir, Stanimirović S. Aleksandar		
Lecturer/associate (for exercises)		Davidović P. Nikola		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		The aim of the course is to introduce students to advanced distributed algorithms and data structures, which enable an understanding of the concept of blockchain systems, as well as to advanced technologies that enable the implementation of these systems.		
Course outcomes		The student is expected to acquire the necessary knowledge to understand the principles and concepts related to blockchain technology, as well as to be able to apply available tools and environments, and implement private and public blockchain systems independently or in available environments. The student is expected to master the technologies in order to be able to independently develop distributed applications on the Ethereum blockchain platform.		
Course outline				
Theoretical teaching		Basic blockchain concepts. Decentralization. Communication in decentralized systems. Elements of cryptography, cryptographic primitives. Public key systems, PKI, RSA, ECC. Hash functions, SHA-1, SHA-2, MD5. Using OpenSSL. Fault tolerance and consensus algorithms. Cryptocurrencies and cryptoeconomics. Public and private blockchain systems. Bitcoin. Digital keys and addresses. Transactions. Mining, CPU, GPU, FPGA, ASIC. Alternative currencies. Ethereum. Smart contracts. Programming in Ethereum. Implementation of the Ethereum system. Development tools. Hyperledger. Implementation of blockchain systems.		
Practical teaching (exercises, OFE, study and research work)		Interacting with a digital wallet. The difference between the main Ethereum network and the test network. Programming in the Solidity programming language. Programming a smart contract. Remix development environment. Basics of the Node.js programming language and the npm package manager. Basics of Javascript ES6 syntax. Truffle tools. Deploying a smart contract on the blockchain. Web3.js and interacting with a smart contract. Testing smart contracts. Basics of React web technology. Creating a web application as a user interface for a distributed application.		
Textbooks/references				
	1	Imran Bashir, "Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained", 2nd Edition, ISBN-10: 978-1788839044, 2018.		
	2	Andreas Antonopoulos, Gavin Wood, "Mastering Ethereum: Building Smart Contracts and DApps", ISBN-10: 978-1491971949, 2019.		
	3	Chris Dannen, "Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners", 1st. edition, ISBN-10: 978-1484225349, 2017.		
	4	Arshdeep Bahga, Vijay Madisetti, "Blockchain Applications: A Hands-On Approach", ISBN-10: 978-0996025553, 2017.		
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, Auditive exercises, Laboratory exercises. Student project realization.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures	10	Written exam		
Exercises	10	Oral exam		40
Colloquia				
Projects	40			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Computer Based Sensor Systems		
Lecturer (for lectures)		Denić B. Dragan, Dinčić R. Milan		
Lecturer/associate (for exercises)		Dinčić R. Milan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	To familiarize students with the fundamental principles, significance, and applications of sensors and sensor systems; techniques for data acquisition, processing, and analysis; modern sensor technologies, including telemetry systems, wireless sensor networks, and Internet of Things (IoT) solutions; programming in LabVIEW; and the development of sensor systems using contemporary hardware-software platforms such as LabVIEW, Raspberry Pi, and FPGA.			
Course outcomes	Students will acquire both theoretical and practical knowledge of modern sensor technologies and systems. Additionally, they will gain skills in hardware and software implementation of sensor systems using LabVIEW, Raspberry Pi, and FPGA platforms.			
Course outline				
Theoretical teaching	Sensors and Sensor Systems. Principles of operation, importance, and applications of sensors. General structure of computer-based sensor systems. Data acquisition. Acquisition cards. Protocols for connecting acquisition cards to a computer. Computer software for data acquisition. LabVIEW software package – features, applications, programming. Concept of virtual instrumentation. Implementation of sensor systems based on LabVIEW, Raspberry Pi, and FPGA platforms. Analysis and processing of measurement data. Telemetry systems. Wireless sensor networks. IoT systems.			
Practical teaching (exercises, OFE, study and research work)	Problem-solving exercises, practical exercises, laboratory work, project assignments, and seminar papers, with the aim of enabling students to acquire practical knowledge related to programming in LabVIEW and to the design and implementation of sensor systems using LabVIEW, Raspberry Pi, and FPGA technologies.			
Textbooks/references				
1	Behzad Ehsani, "Data Acquisition using LabVIEW", Packt Publishing, 2017.			
2	Maurizio Di Paolo Emilio, "Data Acquisition Systems: From Fundamentals to Applied Design", Springer-Verlag, 2013.			
3	Robert Bishop, "Learning With LabVIEW", Pearson, 2015.			
4	National Instruments, "Data Acquisition and Signal Conditioning Course Manual", 2012.			
5	John Shovic, "Raspberry Pi IoT Projects", 2015.			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, problem-solving exercises, laboratory exercises, discussion of student solutions to assigned project tasks, consultations.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures	5	Written exam		25
Exercises	20	Oral exam		25
Colloquia	25			
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering	
Type and level of studies		Master Academic Studies	
The name of the course		DevOps - Integrated Approach to Software Development	
Lecturer (for lectures)		Predić B. Bratislav	
Lecturer/associate (for exercises)		Predić B. Bratislav	
Lecturer/associate (for OFE)			
Number of ECTS	6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None		
Course objectives	The goal of this course is for students to acquire advanced knowledge and skills in the field of DevOps, including process automation, infrastructure as code management, continuous integration and software delivery, monitoring, and security. Through theoretical lectures and practical exercises, students will learn how to improve software development and operational processes, reducing delivery time while increasing the quality and security of applications.		
Course outcomes	Upon successful completion of the course, students will be able to: - Understand the core concepts and principles of the DevOps methodology. - Implement automation processes for software development, testing, and delivery. - Use tools for continuous integration and delivery (CI/CD). - Manage infrastructure using Infrastructure as Code (IaC). - Implement system monitoring and logging. - Optimize application performance and security. - Develop strategies for container orchestration and microservices architectures.		
Course outline			
Theoretical teaching	The following topics provide a balance between theoretical foundations and practical applications, enabling students to gain a comprehensive understanding of DevOps: 1. Introduction to DevOps 2. DevOps Culture and Principles 3. Continuous Integration (CI) 4. Continuous Delivery and Deployment (CD) 5. Version Control and Collaboration 6. Infrastructure as Code (IaC) 7. Containerization and Docker 8. Container Orchestration: Kubernetes 9. Monitoring and Logging 10. Security in DevOps 11. Cloud Computing and DevOps 12. Serverless Architectures and Function as a Service (FaaS) a. Lambda, Azure Functions, Google Cloud Functions b. Benefits and Challenges of Serverless 13. Testing in a DevOps Environment 14. Incident Management and System Resilience 15. DevOps in Practice: Case Study		
Practical teaching (exercises, OFE, study and research work)	The exercises cover key practical aspects of DevOps and prepare students for work in production environments: 1. Working with Git and GitHub – Basic commands, branching, merging, and pull requests. 2. Setting up a Jenkins CI/CD Pipeline – Automating builds, testing, and deployment. 3. Creating Docker Containers – Writing Dockerfiles, building, and running containers. 4. Container Orchestration with Kubernetes – Deployment, scaling, and service management. 5. Using Terraform for Infrastructure Management – Defining cloud infrastructure as code. 6. Implementing Monitoring with Prometheus and Grafana – Collecting and visualizing metrics. 7. Security in DevOps – Working with Vault, container scanning, and secrets management. 8. AWS Lambda and Serverless Functions – Creating and executing serverless functions. 9. Automated Application Testing – Setting up Selenium tests in the CI/CD process. 10. Implementing Logging with the ELK Stack – Configuring and analyzing logs in Elasticsearch.		
Textbooks/references			
1	Kim, G., Behr, K., & Spafford, G. (2013). The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win. IT Revolution Press. ISBN: 978-0988262591		

2	Kim, G., Humble, J., Debois, P., & Willis, J. (2016). The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations. IT Revolution Press. ISBN: 978-1942788003			
3	Forsgren, N., Humble, J., & Kim, G. (2018). Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations. IT Revolution Press. ISBN: 978-1942788331			
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, auditory exercises, laboratory exercises, and student projects.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	30
Exercises		10	Oral exam	40
Colloquia				
Projects		20		

Specification for the book of courses

Study program		Computer Science and Informatics		
Type and level of studies		Master Studies		
The name of the course		Master Thesis – Study and Research Work		
Lecturer (for lectures)				
Lecturer/associate (for exercises)				
Lecturer/associate (for OFE)				
Number of ECTS	6	Course status (obligatory/elective)	Obligatory	
Prerequisites				
Course objectives	Application of basic, theoretical-methodological, scientific-professional and professional-applied knowledge and methods to solve specific problems. The student studies the problem, its structure and complexity and, based on the conducted analyses, draws conclusions about possible ways of solving it. By studying the literature, the student becomes familiar with the methods intended for solving similar tasks and engineering practice in solving them.			
Course outcomes	Enabling students to independently apply previously acquired knowledge from various areas they have studied, in order to observe the structure of a given problem and its systemic analysis in order to draw conclusions about possible directions for its solution. Through the independent use of literature, students expand their knowledge by studying various methods and works related to similar issues. In this way, students develop the ability to conduct analyses and identify problems within the given issue. Through the practical application of acquired knowledge, students develop the ability to perceive the place and role of engineers in the chosen area, the need for cooperation with other professions and teamwork.			
Course outline				
Theoretical teaching	It is formed individually in accordance with the needs of a specific diploma-master's thesis, its complexity and structure. The student, according to his affinities and preferences, chooses the field of study work, or rather the subject teacher from the list of teachers in the study program who defines a specific task for him. The student studies professional literature, professional and scientific works that deal with similar topics, performs analyses in order to find a solution to a specific task or performs certain experiments in the laboratory. The study work also includes active monitoring of primary knowledge, organization and performance of experiments, numerical simulations and statistical data processing, and preparation of a seminar paper in a narrow scientific and educational field to which the topic of the independent research work belongs.			
Practical teaching (exercises, OFE, study and research work)				
Textbooks/references				
1				
2				
3				
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
			12	
Teaching methods				
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	50
Colloquia				
Projects		50		

Specification for the book of courses

Study program		Computer Science and Informatics	
Type and level of studies		Master Studies	
The name of the course		Professional Practice	
Lecturer (for lectures)			
Lecturer/associate (for exercises)			
Lecturer/associate (for OFE)			
Number of ECTS	3	Course status (obligatory/elective)	Obligatory
Prerequisites			
Course objectives	Familiarization with the work process in the company where the professional internship is conducted, its objectives, and organizational units. Getting to know the team and the project that the student joins as part of their professional internship, selected in accordance with the elective study area (module) chosen by the student. Understanding the company's workflow, business processes, work-related risks, and participating in design, documentation development, or quality control, in alignment with the work process and the possibilities of the work environment.		
Course outcomes	Enhancing the student's ability to integrate into the workforce upon completing their studies. Developing responsibility, a professional approach to work, and teamwork communication skills. Supplementing the theoretical knowledge acquired in the academic program with practical insight into the issues studied within the student's field. Utilizing the expertise of professionals employed at the institution where the internship is conducted to broaden practical knowledge and motivate students. Gaining a clear understanding of how the acquired knowledge and skills from the academic program can be applied in practice.		
Course outline			
Theoretical teaching			
Practical teaching (exercises, OFE, study and research work)	The content of the professional practice is fully aligned with its objectives. The student becomes familiar with the company's structure and business goals, adapts their involvement to the chosen field of study, and duly fulfills work responsibilities in accordance with the duties of employees in the company. The student describes their own engagement during the internship and provides a critical reflection on their experience, knowledge, and skills acquired throughout the practice.		
Textbooks/references			
	1		
	2		
	3		
	4		
	5		
Number of classes of active education per week during semester/trimester/year			
Lectures	Exercises	OFE	Study and research work
			Other classes
			6
Teaching methods	The student generally independently selects a company from the public, private, or state sector in which they will complete their professional internship. The internship can also be carried out abroad, in which case the student additionally improves their foreign language skills. Based on the student's proposal, the head of the selected field/module approves the internship at the desired institution and, upon request, issues a written referral for the internship to the person responsible for conducting the practice at the given institution. Upon completion of the internship, and based on the student's report and confirmation from the responsible individual—verified with a signature and the company seal—the student is awarded 3 ECTS credits for the completed practice.		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises	70	Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program		Computer Science and Informatics	
Type and level of studies		Master Studies	
The name of the course		Master Thesis	
Lecturer (for lectures)			
Lecturer/associate (for exercises)			
Lecturer/associate (for OFE)			
Number of ECTS	9	Course status (obligatory/elective)	Obligatory
Prerequisites			
Course objectives	The Master's thesis aims to consolidate, confirm and practically apply the knowledge acquired during the Master's academic studies. The student is given the opportunity to demonstrate the ability to independently carry out a project, which can be of a practical, research or theoretical-methodological nature. The student also gains experience in presenting his/her work through a written form and an oral presentation during the thesis defense.		
Course outcomes	Ability to lead an independent project, ability to formulate and analyze problems, critically review possible solutions, review literature in a given field. Application of acquired engineering and design knowledge and skills to solve problems, taking into account the complexity, costs, reliability and efficiency of the solution. Ability to write a paper in a given form. Ability to clearly explain the completed project through an oral defense of the paper.		
Course outline			
Theoretical teaching	A master's thesis is an independent research, practical or theoretical methodological work of a student in accordance with the level of study, in which he becomes acquainted with a narrow field through a literature review and adopts the research methodology, or design, necessary for the preparation of the thesis. Through the preparation of the thesis, the student applies practical and theoretical knowledge acquired during his studies. The written thesis usually contains an introductory chapter, a definition of the problem, a review of the field and existing solutions, a proposal and description of the solution, a conclusion and literature. A public oral defense of the thesis is organized before a committee of three members, one of whom is the thesis mentor. During the oral defense, the candidate explains the results of his work, and then answers questions from the committee members, thereby demonstrating the candidate's ability to orally present his project.		
Practical teaching (exercises, OFE, study and research work)			
Textbooks/references			
1			
2			
3			
4			
5			
Number of classes of active education per week during semester/trimester/year			
Lectures	Exercises	OFE	Study and research work
			2
Teaching methods			
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	70
Exercises		Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Computer Simulation		
Lecturer (for lectures)		Vučković V. Vladan		
Lecturer/associate (for exercises)		Vučković V. Vladan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Mastering the basic knowledge necessary for the use of basic processes and procedures in computer modeling and simulation.			
Course outcomes	Theoretical knowledge: Mastering mathematical techniques for computer modeling and simulation; 3D modeling and programming of computer simulations.			
Course outline				
Theoretical teaching	Fundamentals of computer simulation. General approach to simulation systems. Mathematical foundations of simulation systems. Mathematical modeling of physical processes and sensor data. Stochastic and deterministic approaches in simulation. Continuous and discrete simulations. Efficient data structures and simulation algorithms. Approach via differential algebraic equations. Local and distributed simulations. Discrete event simulation (DES). Aggregate Level Simulation Protocol (ALSP), Distributed Interactive Simulation (DIS), the High Level Architecture (simulation) (HLA). Introduction to 3D software. 3D simulations in real time. Parallel algorithms in simulation systems. Hardware optimization for simulation machines.			
Practical teaching (exercises, OFE, study and research work)	Hierarchies in 3D software and expression animation. Practical 3D modeling on the computer. Use of modern 3D modeling software. Basic simulation procedures. Keyframe animation – rotations and sizes (squash and stretch). NURBS modeling - lines. NURBS modeling - shapes. Introduction to polygonal modeling. Quad modeling. Polygonal modeling of a simple character. Polygonal character modeling by task. Torso. Polygonal character modeling by task. Head: eyes, lips, ear. UV mapping. Digital sculpting. Basics of 3D materialization. Rigging – Introduction to skeletal systems. The concept of Set Driven Key. Creating and selecting attributes. Programming various simple simulator classes.			
Textbooks/references				
1	Modeling and Simulation in Engineering, Edited by Catalin Alexandru, ISBN 978-953-51-0012-6, Publisher: InTech, Published: March 07, 2012 under CC BY 3.0 license.			
2	B. P. Zeigler, A. Muzy, and E. Kofman, Theory of Modeling and Simulation: Discrete Event & Iterative System Computational Foundations, 3rd ed. London: Academic Press, 2018			
3				
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, blackboard exercises, laboratory exercises, independent student work on homework and projects, consultations.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	30	Oral exam		40
Colloquia	10			
Projects	20			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Application of Multiple Valued Logic in Representation and Processing of Digital Signals		
Lecturer (for lectures)		Radmanović M. Miloš, Rajković J. Petar		
Lecturer/associate (for exercises)		Radmanović M. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS	6	Course status (obligatory/elective)	Obligatory / Elective	
Prerequisites	None			
Course objectives	Students will acquire theoretical and practical knowledge in filed of multiple-valued logic and its application in digital system design and signal processing.			
Course outcomes	Students will learn fundamentals of multiple-valued logic, related algebraic structures, and multiple-valued functions, their representation and implementation on the different technological platforms. They will learn fundamental methods for calculation of multiple-valued functions for use in digital system design and signal processing.			
Course outline				
Theoretical teaching	1. Multiple-valued functions, 2. Applications of multiple-valued functions, 3. Functional expansions for multiple-valued functions, 4. Spectral representations for multiple-valued functions, 5. Decision diagrams for multiple-valued functions, 6. Optimization of decision diagrams, 7. GF decision diagrams, 8. KGF decision diagrams, 9. RMF decision diagrams, 10. WC decision diagrams, 11. Fast algorithms for computing spectral transforms of multiple-valued functions, 12. Algorithms for computing transforms on mCPU, 13. Algorithms for computing transforms on GPU, 14. Algorithms for computing transforms on hybrid platforms.			
Practical teaching (exercises, OFE, study and research work)	Introduction and application of specialized software for working with multiple-valued functions, program implementation for working with multiple-valued functions on graphics processors. Demonstration exercises: 1. Multiple-valued functions, 2. Functional expansions for multiple-valued functions, 3. Spectral representations for multiple-valued functions, 4. Decision diagrams for multiple-valued functions, 5. Optimization of decision diagrams, 6. GF decision diagrams, 7. KGF decision diagrams, 8. RMF decision diagrams, 9. WC decision diagrams, 10. Fast algorithms for calculating spectral transformations of multiple-valued functions, 11. Algorithms for calculating transformations on mCPU, 12. Algorithms for calculating transformations on GPU, 13. Algorithms for calculating transformations on hybrid platforms.			
Textbooks/references				
1	R.S. Stankovic, J.Astola, C. Moraga, Representations of Multiple- Valued Logic Functions , Morgan & Claypool Publishers, 2012.			
2	D. Michael Miller, Mitchell Thornton, Multiple Valued Logic: Concepts and Representations, Morgan & Claypool Publishers, 2008.			
3	G. Karniadakis, R.M. Kirby, Parallel Scientific Computing in C++ and MPI: A Seamless Approach to Parallel Algorithms and Their Implementation, Cambridge University Press, 2003.			
4	S. Cook, CUDA Programming: A Developer's Guide to Parallel Computing with GPUs, Morgan Kaufmann, 2012.			
5	Teaching material on the website: http://cs.elfak.ni.ac.rs/nastava/			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures and demonstrations using slides. Seminars and projects.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises		Oral exam		40
Colloquia				
Projects	60			

Specification for the book of courses				
Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Semantic Web		
Lecturer (for lectures)		Tošić B. Milorad, Nejković M. Valentina		
Lecturer/associate (for exercises)		Nejković M. Valentina		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		Information Systems, Web Programming, Databases, and Artificial Intelligence.		
Course objectives		Acquisition of basic theoretical knowledge and potential application areas of the Semantic Web. Mastering fundamental programming techniques for developing Semantic Web applications at the current stage of technology development. Fostering a creative attitude among students toward possible directions for further technological advancement in this field.		
Course outcomes		A systematic approach to the application of the Semantic Web in current areas of use has been developed and adopted. Theoretical knowledge of semantics in information technologies has been acquired. Students understand what ontologies are and are capable of working with them practically, as well as applying them in the currently most attractive web applications.		
Course outline				
Theoretical teaching		Introduction: Structure, syntax, and semantics. The need for semantics on the Web. Meta-programming: Metadata, XML Schema; XSLT. RDF. RDFS. Semantics: Semantics and knowledge; Ontologies; Logics; Reasoning; Domain modeling; Context. Distributed knowledge: Classification; Knowledge-based protocols. Technologies: Tools for working with ontologies; Programming packages (APIs) for ontology management; OWL. Methodologies: Ontology engineering methodologies; Methodologies for developing semantic systems. SPARQL query language and protocol. SHACL language for constraints in SPARQL. Automated reasoning. Reasoning using SPARQL query language. Semantic systems: Semantic Web Services, Semantic Web Portals, Semantic Multi-Agent Systems. Development of applications based on knowledge graphs: Searching and analyzing large datasets, Intelligence in business applications, Intelligence in integration and development of Web applications.		
Practical teaching (exercises, OFE, study and research work)		Introduction to structure, syntax, and semantics. Exercises on XML syntax. Structuring data on the Web. Practical examples of using semantics for improved search. Differences between structured and unstructured data with examples. Meta-programming and metadata. Defining and using metadata in XML. Examples with XML schemas. XSLT transformations. Using XSLT for data presentation. Converting XML data to HTML format. RDF (Resource Description Framework). Creating RDF graphs. Practical use of RDF for resource description. Ontologies and logics. Structuring ontologies using RDF and RDFS. Introduction to logical reasoning systems. Reasoning and domain modeling. Using logical rules for reasoning. Domain modeling with ontologies. Context and distributed knowledge. Modeling context in semantic applications. Classification of distributed knowledge. Tools and APIs for working with ontologies. Introduction to the Protégé tool. Using programming packages for ontology manipulation. OWL (Web Ontology Language). Creating ontologies in OWL. Linking OWL ontologies with RDF data. Semantic systems. Practical applications of the Semantic Web in multi-agent systems. Development of semantic web services and portals. Applications and data analysis. Searching and analyzing large datasets. Examples of intelligence in business applications and web integration.		
Textbooks/references				
	1	Arp, Robert, Barry Smith, and Andrew D. Spear. Building ontologies with basic formal ontology. Mit Press, 2015.		
	2	Dietz, Jan L. G., and Hans B. F. Mulder. Enterprise Ontology: A Human-Centric Approach to Understanding the Essence of Organisation. 2nd ed. Springer, 2024.□		
	3	Keet, C. Maria. "An introduction to ontology engineering." (2018).		
	4	DuCharme, Bob. Learning SPARQL: querying and updating with SPARQL 1.1. " O'Reilly Media, Inc.", 2013.		
	5	Allemang, Dean, James Hendler, and Fabien Gandon. Semantic Web for the Working Ontologist: Effective Modeling for Linked Data, RDFS, and OWL. 3rd ed. ACM Books, 2020.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes

2	2	0	0	0
Teaching methods	Lectures; Auditory exercises; Laboratory exercises; Computer exercises; Consultations; Independent student research; Oral presentations by students on chosen/assigned topics; Active student participation in classes through the use of the interactive course website.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam	Points	
Activity during lectures	30	Written exam		
Exercises	30	Oral exam	40	
Colloquia				
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Methods and Techniques of the Image Processing		
Lecturer (for lectures)		Vučković V. Vladan		
Lecturer/associate (for exercises)		Radmanović M. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Acquiring knowledge about image processing techniques and training students to independently apply the learned techniques to solve real-world problems in the field of image processing.			
Course outcomes	Upon completion of this course, students should become familiar with the methods and techniques used in the field of image processing and acquire sufficient knowledge to independently implement applications based on image processing.			
Course outline				
Theoretical teaching	Principles of image digitization. Image quantization in the original and transformation domains. Statistical image and noise models. Image transformations. Discrete representations of image transformations. Methods for filtering in the original and transformation domains. Quantification. Image restoration: Linear filters. Image restoration and segmentation: nonlinear filters. Methods for improving image quality. Image compression. Efficient image processing algorithms. Processing scanned documents. OCR systems.			
Practical teaching (exercises, OFE, study and research work)	Study research work: preparation of seminar papers in the field of advanced image processing techniques and oral presentation and defense of the paper.			
Textbooks/references				
1	L. Yaroslavsky, Digital Holography and Digital Image Processing: Principles, Methods, Algorithms, Kluwer Academic, 2004.			
2	Yaroslavsky, L., Advanced Lab in Image Processing, teaching material based on the book Digital Holography and Digital Image Processing: Principles, Methods, Algorithms, Kluwer Academic, 2004.			
3	B. Jahne, Digital Image Processing, Springer, 2002.			
4	Material on the website: http://cs.elfak.ni.ac.rs/nastava/			
5	R. C. Gonzalez and R. E. Woods, Digital Image Processing, 4th ed. Hoboken, NJ: Pearson, 2018.			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, independent work of students on homework and projects, consultations.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	20	Oral exam		50
Colloquia				
Projects	30			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Medical Information Systems		
Lecturer (for lectures)		Janković S. Dragan, Milenković M. Aleksandar		
Lecturer/associate (for exercises)		Milenković M. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The course objectives are introducing students in specificities of medical information system development, deployment and project management. The students will get the knowledge about different types of medical software as well as related standards and legislature.			
Course outcomes	After the course completion, the students will be able to participate in medical information system development process using theoretical and practical knowledge obtained through lectures and software project development.			
Course outline				
Theoretical teaching	Introduction to medical information systems. Data quality . Standards and legislature. Comparison of existing systems . Medical information system design specificities. Aspects of system implementation . System and data security . Overview on implementation technologies. System installation and deployment phase . System exploitation phase. System interoperability. Telemedicine. Decision support systems. Medical information system project management			
Practical teaching (exercises, OFE, study and research work)	Defining main elements of the future information system. Designing database structure and electronic patient record. Medical examination support software components. Patient scheduling software module. Report generation module. Privileges within medical information system. Data security. Designing the data access Web service. Personal health Web portals. Composite electronic health records. Document management systems. Medical information system taxonomy. Data exchange standards. Data synchronization techniques			
Textbooks/references				
1	Alfred Winter, Elske Ammenwerth, Reinhold Haux, Michael Marschollek, Bianca Steiner, Franziska Jahn, Health Information Systems: Technological and Management Perspectives, Springer, 202			
2	Joseph Tan, Fay Cobb Payton: Adaptive Health Management Information Systems: Concepts, Cases, and Practical Applications, Third Edition, Jones & Bartlett Publishers; 3 edition (May 21, 2009), ISBN-10: 0763756911			
3	Dean F. Sittig, Joan S. Ash: Clinical Information Systems: Overcoming Adverse Consequences (Jones and Bartlett Series in Biomedical Informatics), ones & Bartlett Publishers; 1 edition (November 23, 2009), ISBN-10: 0763757640			
4	Scott Coplan, David Masuda: Project Management for Healthcare Information Technology, McGraw-Hill Professional; 1 edition (February 1, 2011), ISBN-10: 0071740538			
5	Jingsong Li, Yu Tian, Tianshu Zhou, Healthcare Information Systems: Progress, Challenges and Future Directions, Springer Singapore, 2024.			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, Auditive exercises, Laboratory exercises. Student project realization.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		30
Exercises		Oral exam		20
Colloquia				
Projects	50			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Advanced Learning Technologies		
Lecturer (for lectures)		Milentijević Z. Ivan		
Lecturer/associate (for exercises)		Vojinović M. Oliver		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	To enable students to acquire knowledge about contemporary approaches to learning and education, the potential for enhancing learning through information technologies, and the pedagogical and cognitive effects of technology integration in learning.			
Course outcomes	Upon successful completion of the course, students will be able to analyze specific educational objectives and environments, and design technological solutions to support learning. They will gain practical experience in selecting, customizing, and developing components and systems for learning support.			
Course outline				
Theoretical teaching	Learning paradigms – instructivist and constructivist approaches. Instructional design and the use of technological tools in instructional design. Formal and informal learning environments: games, e-learning, computer-supported collaborative learning, problem- and project-based learning, and social interaction-based learning. Computer-assisted testing – generic, adaptive, and standardized testing. Standard formats for presenting learning materials and curricula. Supporting learning with AI. Advanced learning analytics with models.			
Practical teaching (exercises, OFE, study and research work)	Analysis of learning approaches, educational objectives, and learning environments, along with mapping appropriate technologies. Design and development of software tools to support learning.			
Textbooks/references				
	1	Sanju Saha, Santoshi Halder, The Routledge Handbook of Education Technology, Taylor&Francis, 2023.		
	2	Yang Shen, Xin Yin, Yu Jiang, Lingkai Kong, Sheng Li, Haijun Zeng, Case Studies of Information Technology Application in Education: Utilising the Internet, Big Data, Artificial Intelligence, and Cloud in Challenging Times, Springer, 2023.		
	3	Jim Goodell, Learning Engineering Toolkit: Evidence-Based Practices from the Learning Sciences, Instructional Design, and Beyond, Routledge, 2022.		
	4	A curated collection of selected accademic articles		
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, consultations, students' work on assignments and projects, seminars (presentations of student projects, with discussion).			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		10	Written exam	
Exercises		10	Oral exam	40
Colloquia				
Projects		40		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Spectral Techniques		
Lecturer (for lectures)		Radmanović M. Miloš		
Lecturer/associate (for exercises)		Radmanović M. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The main goal of the course is for students to acquire theoretical and practical knowledge in the field of application of spectral techniques in the design, analysis and implementation of digital systems.			
Course outcomes	Students will be introduced to various techniques for representing discrete functions, techniques and methods for calculating discrete transformations, and some applications of spectral techniques in the analysis and implementation of digital systems.			
Course outline				
Theoretical teaching	1. Representation of discrete functions, 2. Functional expansions, 3. Decision diagrams, 4. Spectral transformations of logical functions, 5. Techniques and methods for calculating spectral transformations, 6. Fast algorithms, 7. Walsh spectrum, 8. Haar spectrum, 9. Vilenkin–Chrestenson spectrum. 10. Calculation of discrete transformations using decision diagrams, 11. Spectral methods for the synthesis of logic networks. 12. Synthesis of sequential networks using spectral methods, 13. Spectral methods for testing digital systems.			
Practical teaching (exercises, OFE, study and research work)	Work with specialized software packages. Program implementation of algorithms for calculating spectral transformations and their application. Demonstration exercises: 1. Functional developments, 2. Decision diagrams, 3. Spectral transformations of logical functions, 4. Techniques and methods for calculating spectral transformations, 5. Fast algorithms, 6. Walsh spectrum, 7. Haar spectrum, 8. Vilenkin–Chrestenson spectrum. 9. Calculation of discrete transformations using decision diagrams, 10. Spectral methods for synthesizing logic networks. 11. Synthesis of sequential networks using spectral methods, 12. Spectral methods for testing digital systems.			
Textbooks/references				
1	M. Karpovsky, R. Stankovic, J. Astola, Spectral Logic and Its Applications for the Design of Digital Devices, John Wiley & Sons, 2008.			
2	M. A. Thornton, R. Drechsler, D. M. Miller, Spectral Techniques in VLSI CAD, Kluwer Academic Publishers USA, 2012.			
3	R. Drechsler, B. Becker, Decision Diagrams: Theory and Implementation, Springer USA, 2013.			
4	R. Ebendt, G. Fey, and R. Drechsler, Advanced BDD Optimization. Dordrecht, Netherlands, Springer, 2005.			
5	Teaching material on the website: http://cs.elfak.ni.ac.rs/nastava/			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures and demonstrations using slides, seminars and projects.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises		Oral exam		40
Colloquia				
Projects	60			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Advanced Techniques in 3D Modeling and Animation		
Lecturer (for lectures)		Vučković V. Vladan		
Lecturer/associate (for exercises)		Vučković V. Vladan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Mastering the basic knowledge necessary for the use of advanced processes and procedures in computer 3D modeling and animation.			
Course outcomes	Theoretical knowledge: Mastering advanced techniques for computer 3D modeling and animation; 3D modeling, programming camera paths, and generating animations on the computer.			
Course outline				
Theoretical teaching	3D Layout - scenes in 3D software. Fundamentals of rendering. Mental Ray – Antialiasing, GI, Final Gather. Mental ray Shaders: Mia X Pass, SSS, Illumination Shaders, Displacement, Ambient Occlusion. Mental Ray – Render Layers, Render Passes, Contribution maps. Hardware render. Lighting. Rendering optimization. Fundamentals of computer animation. Rendering animation. Modeling camera paths. Digital animation direction. Planning and organization of a digitally generated and animated film project. Technical development of the project.			
Practical teaching (exercises, OFE, study and research work)	Exercises; Preparation of seminar papers. Introduction to advanced modeling and animation methods. Practical 3D modeling and animation on the computer. Use of modern 3D animation software. Generating various simple animations.			
Textbooks/references				
1	"Computer Animation, Second Edition: Algorithms and Techniques", Morgan Kaufmann; 2 edition (October 11, 2007); ISBN-10: 0125320000 ISBN-13: 978-0125320009			
2	J. Vince, Introduction to Computer Animation, 4th ed. London: Springer, 2019.			
3	E. Lengyel, Mathematics for 3D Game Programming and Computer Graphics, 3rd ed. Boca Raton, FL: CRC Press, 2019.			
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, blackboard exercises, laboratory exercises, independent student work on homework and projects, consultations.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		30	Oral exam	40
Colloquia		10		
Projects		20		

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Intelligent Information Systems		
Lecturer (for lectures)			Tošić B. Milorad		
Lecturer/associate (for exercises)			Bogdanović D. Miloš		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Obligatory / Elective
Prerequisites		Passed exams in Information Systems, Web Programming, and Artificial Intelligence.			
Course objectives		Acquisition of practical programming skills, theoretical knowledge, and a systematic approach necessary for the design, implementation, and operation of systems in which information technologies, computers, the Internet, and humans collaborate to achieve results characterized as intelligent.			
Course outcomes		Students are able to analyze, design, and implement intelligent information systems in various domains and identify appropriate methods and technologies for their development. They are capable of applying semantic technologies, ontologies, language models, and intelligent agents, as well as integrating and retrieving information from diverse sources. They possess practical skills for developing service-oriented, multi-agent, and language-model-based applications.			
Course outline					
Theoretical teaching		General conceptual assumptions: data models, information, and knowledge; fundamental technologies (Internet, databases, artificial intelligence, information retrieval); business aspects; mathematical foundations. Intelligent databases, deductive and active databases, intelligent information retrieval. Reasoning. Web. The Web and intelligent information systems: Ontologies, Linked Data, Semantic Web, Social Networks, Collaborative Systems, Extensible and Small-World Networks. End-user programming. Architectures of intelligent information systems: service-oriented and multi-agent architectures. Knowledge-based systems. Information retrieval and browsing: Web agents, Web data collection ("crawling"), intelligent Web tools, Web search engines, and the semantics problem. Language models. Development of applications based on language models. Query engineering for language models. Interaction models in systems with intelligent agents based on language models. Data flow models. Development of multi-agent applications. Development of conversational robots (chatbots) on a given topic. Information management, information flow, integration of structured and semi-structured information sources, XML- and RDF-based methods for information representation.			
Practical teaching (exercises, OFE, study and research work)		Tools for ontology development. Programming interfaces for creating RDF knowledge graphs in Java and Python programming languages. SPARQL queries and reasoning in semantic knowledge graphs. Examples in business information systems. Development of prototypes for semantic-enabled intelligent systems. Integration of intelligent systems with tools for automatic data collection from the Web. Tools for summarizing textual and visual data. Application of large language models in intelligent systems. Use of large language models in ontology learning and automation of semantic systems (automatic query generation). Use of multimodal summarization in intelligent systems. Application of Retrieval Augmented Generation (RAG) for information retrieval in intelligent systems. Predictive methods in intelligent information systems. Management of computational infrastructures in intelligent systems – DevOps, DevSecOps, and MLOps tools.			
Textbooks/references					
1		Romero, José Raúl, Inmaculada Medina-Bulo, and Francisco Chicano, eds. Optimising the software development process with artificial intelligence. Springer, 2023.			
2		Huyen, Chip. Designing machine learning systems. " O'Reilly Media, Inc.", 2022.			
3		Phoenix, James, and Mike Taylor. Prompt Engineering for generative AI. " O'Reilly Media, Inc.", 2024.			
4		Fensel, Dieter, Umutcan Simsek, Kevin Angele, Elwin Huaman, Elias Kärle, Oleksandra Panasiuk, Ioan Toma, Jürgen Umbrich, and Alexander Wahler. Knowledge graphs. Switzerland: Springer International Publishing, 2020.			
5		Dietz, Jan LG, and Hans BF Mulder. Enterprise ontology: A human-centric approach to understanding the essence of organisation. Springer Nature, 2024.			
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes

2	2	0	0	0
Teaching methods	Lectures, exercises, programming assignments			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		30	Written exam	
Exercises		30	Oral exam	40
Colloquia				
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Information Technologies for Development of E-Government Systems		
Lecturer (for lectures)		Stoimenov V. Leonid, Frtunić-Gligorijević B. Milena		
Lecturer/associate (for exercises)		Frtunić-Gligorijević B. Milena		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Introduction to the basic concepts of e-Government systems and mastery of technologies for designing and implementing various aspects of such systems.			
Course outcomes	Theoretical and practical knowledge of the concepts, design and implementation of basic aspects of e-Government systems			
Course outline				
Theoretical teaching	Introduction to the basic concepts of e-Government systems. A brief overview of the history of e-Government system development. Analysis of key aspects of e-Government: data, e-Services, applications, infrastructure. Data management in e-Government systems: models and databases, data mining techniques, ontologies, semantic analysis. Open data and open data platforms. Analysis of e-Government open data portals. Aspects of information integration: service-oriented architecture, Web services. Development of back-office and front-office applications. Current issues in the development of e-Government infrastructure: cloud technologies. Data storage, storage and management, the concept of Big Data, databases, Data warehouse, Data Lake and other concepts. The place of e-Government in the concept of a Smart City.			
Practical teaching (exercises, OFE, study and research work)	Introduction to open platforms for viewing and storing e-government data. Analysis of open data formats. Open data standards. Semantic analysis and discovery of new knowledge by analyzing existing data. Access to open data portals via API interfaces for data search and analysis. Analysis of data openness according to defined methodologies. Implementation of services for citizens based on available open data.			
Textbooks/references				
	1	Mahmood, Z. (2013). E-Government Implementation and Practice in Developing Countries (pp. 1-348). doi:10.4018/978-1-4666-4090-0		
	2	Reddick, C. & Aikins, Stephen K. (2012). Political, Policy and Management Implications Series: Public Administration and Information Technology (pp. 1- 275), Vol. 1, Springer.		
	3	Vitvar, T., Peristeras, V., & Tarabanis, K. (2010). Semantic Technologies for E-Government. XVI, (pp. 1 - 320), ISBN 978-3-642-03507-4, Springer.		
	4	Garson, G. D. (2006). Public Information Technology and E-Governance: Managing the Virtual State. Boston: Jones & Bartlett. 2006.		
	5	Milić, P., Veljković, N., & Stoimenov, L. (2018). Semantic technologies in e-government: Toward openness and transparency. In Smart Technologies for Smart Governments (pp. 55-66). Springer, Cham.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, auditory exercises, laboratory exercises, independent work of students on homework and projects			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	40	Oral exam		30
Colloquia				
Projects	30			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Fuzzy Logic		
Lecturer (for lectures)		Radmanović M. Miloš		
Lecturer/associate (for exercises)		Radmanović M. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Students should acquire basic knowledge in the field of Fuzzy Logic and its application in computing, automatic control, and related fields.		
Course outcomes		Upon completion of this course, students should know what Fuzzy systems are, what their characteristics are, how they are described and implemented, and should be able to practically implement simple Fuzzy systems.		
Course outline				
Theoretical teaching		1. Evolution from classical to fuzzy logic, 2. Generating membership functions, 3. T-norms, T-conorms and negations, 4. Fuzzy sets, 5. Logical consistency, 6. Linguistic functions, 7. Probability measures, 8. Fuzzy modeling using “if-then” fuzzy rules, 9. Linguistic modifiers, 10. Aggregation functions, 11. Fuzzy rules, 12. Fuzzy implication analysis and numerical methods for its calculation, 13. Practical applications of fuzzy logic,		
Practical teaching (exercises, OFE, study and research work)		Study research work: seminar papers and independent practical implementation of a simple Fuzzy system. Demonstration exercises: 1. Generating membership functions, 2. T-norms, T-conorms and negations, 3. Fuzzy sets, 4. Logical consistency, 5. Linguistic functions, 6. Probability measures, 7. Fuzzy modeling using “if-then” fuzzy rules, 8. Linguistic modifiers, 9. Aggregation functions, 10. Fuzzy rules. 11. Fuzzy implication analysis and numerical methods for its calculation, 12. Practical applications of fuzzy logic,		
Textbooks/references				
1		M. Mukaidono, Fuzzy Logic for Beginners, World Scientific, 2001.		
2		T. Niimura, T., K. Tanaka, An Introduction to Fuzzy Logic for Practical Applications, Springer New York, 1996.		
3		J. Harris, Fuzzy Logic Applications in Engineering Science, Springer Netherlands, 2011.		
4		Material on the website: Moraga. C., et al. Introduction to Fuzzy Logic, video lectures.		
5		Material on the website: http://cs.elfak.ni.ac.rs/nastava/		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures and demonstrations using slides and interactive computer work. Practical exercises on computers.		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	40
Colloquia				
Projects		60		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Artificial Intelligence Applications in Information Systems		
Lecturer (for lectures)		Tošić B. Milorad		
Lecturer/associate (for exercises)		Petrović N. Nenad		
Lecturer/associate (for OFE)				
Number of ECTS	6	Course status (obligatory/elective)	Obligatory / Elective	
Prerequisites	None			
Course objectives	Introducing students to current artificial intelligence techniques and case studies where they can be applied in information systems.			
Course outcomes	Upon completion of the course, the student should be capable of developing services that rely on artificial intelligence, successfully applying AI techniques within information systems, and integrating them effectively.			
Course outline				
Theoretical teaching	Introduction to machine learning. Machine learning in information systems. Supervised machine learning. Unsupervised machine learning. Applications of supervised and unsupervised machine learning in information systems. Introduction to generative artificial intelligence. Large language models. Application of large language models in information systems. Multimodal summarization. Application of multimodal summarization in information systems. Retrieval Augmented Generation (RAG). Knowledge graphs in information systems. Ontologies and semantic technologies. Use of semantics in information systems. Synergy of semantic technologies and large language models. Explainable artificial intelligence. Overview of techniques and applications of explainable AI in information systems. Ontologies and semantic ontologies in the field of explainable artificial intelligence. AI-driven development of information systems.			
Practical teaching (exercises, OFE, study and research work)	Weka API for Java. Machine learning in Python. Examples of classification and regression in information systems. Examples of clustering in information systems. Application of large language models in Python. Application of visual language models in Python. Development of prototypes and integration of LLM services with information systems. RDF representation of semantic knowledge graphs. SPARQL queries. Examples of using semantic graphs in information systems. APIs and libraries for explainable artificial intelligence. Examples of the use of explainable AI in information systems. Application of generative artificial intelligence in the development of information systems.			
Textbooks/references				
1	Alammar J., Grootendorst M., Hands-On Large Language Models, O'Reilly Media, Inc; 2024.			
2	Hitzler P., Rudolph S., Krötzsch M., Foundations of Semantic Web Technologies, Chapman & Hall/CRC, 2011.			
3	Ebert I., Understanding Machine Learning with Java and Weka: A beginner's Guide to Building Predictive Models and Algorithms with Java, 2024.			
4	Rothman D., Hands-On Explainable AI (XAI) with Python: Interpret, visualize, explain, and integrate reliable AI for fair, secure, and trustworthy AI apps, Packt Publishing, 2020.			
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures; Auditory exercises; Laboratory exercises; Computer exercises; Consultations; Independent student research; Oral presentations by students on chosen/assigned topics; Active student participation in classes through the use of the interactive course website.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures	30	Written exam		
Exercises	30	Oral exam		40
Colloquia				
Projects				

Specification for the book of courses

Study program	Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering
Type and level of studies	Master Academic Studies
The name of the course	Medical Signal and Image Processing and Analysis
Lecturer (for lectures)	Janković S. Dragan
Lecturer/associate (for exercises)	Čikić D. Đorđe
Lecturer/associate (for OFE)	
Number of ECTS	6 Course status (obligatory/elective) Obligatory / Elective
Prerequisites	None
Course objectives	Introduction to signals and images encountered in devices used in medicine and methods for analyzing and processing them.
Course outcomes	Complete familiarization with signals and images encountered in medical devices. Mastery of methods for processing and analyzing signals and images encountered in medicine. Developed a critical attitude towards the limitations and benefits of applying signals and images in medicine.
Course outline	
Theoretical teaching	Introduction to medical signal and image processing: Overview of medical signals (EEG, ECG, EMG) and images (RTG, MRI, CT, ultrasound); Basic mathematical models and digital data representation; Sampling and quantization in medical signals and images; Standards and formats of medical signals and images; DICOM Standard, PNG, JPG; Data visualization. Digital medical image processing: Types of signals and systems, LTI systems; Basic signal operations: filtering (Gaussian filter, Median filter, Sharpening), normalization; Fourier transform, Short-time Fourier transform, Spectrogram, Wavelet transform. Digital medical image processing: Intensity transformations of medical images. Spatial transformations of medical images (filtering). Segmentation, Morphological operations on medical images. Fourier transform, Wavelet transform. Sobel filter, Detection of edges, lines and circles. Textures in medical images. Statistical processing of data, signals and images: ICA and PCA for dimensionality reduction; Removing invalid data. Noise filtering in signals and images: mathematical noise models; Gaussian noise, Impulse noise and filtering methods; Periodic and quasi-periodic noise in medical images. Analysis and processing of biomedical signals: Processing and analysis of EEG signals (detection of epileptic seizures). Processing of ECG signals (detection of arrhythmias and analysis of heart rhythm). Processing of EMG signals (recognition of muscle movements). 3D reconstruction and visualization: Basics of 3D medical data (CT and MRI images); Segmentation of organs and tissues in 3D space. Machine learning in medical data processing: Machine learning models for analyzing medical signals. Clustering of medical images (Fuzzy C means clustering, Superpixels...). Deep learning in medical data processing: Application of neural networks in medical data processing. Application of convolutional neural networks for medical image and signal processing. Automated diagnostics: Computer vision in medical diagnostics. Integration of medical data with medical information systems.
Practical teaching (exercises, OFE, study and research work)	Overview of tools: Python (Numpy, SciNumPy, OpenCV, Scikit-learn). Loading and visualization of medical signals and images. Introduction to tools for visualization of signals and images. Applying different filters to signals. Applying Fourier and Wavelet transforms to signals. Calculating signal spectrograms. Filtering (Gaussian filter, Median filter, Sharpening). Otsu, Watershed and Active contours for segmentation. Morphological operations and Morphological reconstruction. Fourier transform. Wavelet transform. Canny edge detector. Segmentation and Hough transform for nucleus detection. Haralick features for textures in medical images. Applying ICA to remove invalid electrodes in EEG signals. Analysis of EEG signal segments. Applying Median filter to filter impulse noise. Spectral filtering of periodic and quasi-periodic noise. Peak detection in ECG signals. EEG signal analysis using Wavelet transform. Data visualization in VTK/ITK libraries. Loading DICOM series and 3D reconstruction of organs. Generating 3D models from CT scans using VTK. Construction of superpixels on the example of histopathological images. Classification of superpixels using machine learning models. Application of Random Forest and SVM for ECG signal analysis. Data processing for migraine prediction. Introduction to TensorFlow and PyTorch libraries. Implementation of CNN models for classification of X-ray images.
Textbooks/references	
1	Rafael C Gonzalez, Richard E Woods, Digital Image Processing, Pearson Education, 4th edition, 2017.

2	Alan Oppenheim, Alan Willsky,S. Hamid, Signals and systems, Pearson, 2015.			
3	Geoff Dougherty, Digital image processing for medical application, Cambridge University Press, 2009.			
4	Kunal Pal, Samir Ari, Arindam Bit, Advanced Methods in Biomedical Signal Processing, Academic Press, 2022.			
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, Auditive exercises, Laboratory exercises.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	60
Colloquia				
Projects		40		

Specification for the book of courses

Study program		Computer Science and Informatics	
Type and level of studies		Master Studies	
The name of the course		Master Thesis – Study and Research Work	
Lecturer (for lectures)			
Lecturer/associate (for exercises)			
Lecturer/associate (for OFE)			
Number of ECTS	6	Course status (obligatory/elective)	Obligatory
Prerequisites			
Course objectives	Application of basic, theoretical-methodological, scientific-professional and professional-applied knowledge and methods to solve specific problems. The student studies the problem, its structure and complexity and, based on the conducted analyses, draws conclusions about possible ways of solving it. By studying the literature, the student becomes familiar with the methods intended for solving similar tasks and engineering practice in solving them.		
Course outcomes	Enabling students to independently apply previously acquired knowledge from various areas they have studied, in order to observe the structure of a given problem and its systemic analysis in order to draw conclusions about possible directions for its solution. Through the independent use of literature, students expand their knowledge by studying various methods and works related to similar issues. In this way, students develop the ability to conduct analyses and identify problems within the given issue. Through the practical application of acquired knowledge, students develop the ability to perceive the place and role of engineers in the chosen area, the need for cooperation with other professions and teamwork.		
Course outline			
Theoretical teaching	It is formed individually in accordance with the needs of a specific diploma-master's thesis, its complexity and structure. The student, according to his affinities and preferences, chooses the field of study work, or rather the subject teacher from the list of teachers in the study program who defines a specific task for him. The student studies professional literature, professional and scientific works that deal with similar topics, performs analyses in order to find a solution to a specific task or performs certain experiments in the laboratory. The study work also includes active monitoring of primary knowledge, organization and performance of experiments, numerical simulations and statistical data processing, and preparation of a seminar paper in a narrow scientific and educational field to which the topic of the independent research work belongs.		
Practical teaching (exercises, OFE, study and research work)			
Textbooks/references			
	1		
	2		
	3		
	4		
	5		
Number of classes of active education per week during semester/trimester/year			
Lectures	Exercises	OFE	Study and research work
			12
Teaching methods			
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises		Oral exam	50
Colloquia			
Projects	50		

Specification for the book of courses

Study program		Computer Science and Informatics	
Type and level of studies		Master Studies	
The name of the course		Professional Practice	
Lecturer (for lectures)			
Lecturer/associate (for exercises)			
Lecturer/associate (for OFE)			
Number of ECTS	3	Course status (obligatory/elective)	Obligatory
Prerequisites			
Course objectives	Familiarization with the work process in the company where the professional internship is conducted, its objectives, and organizational units. Getting to know the team and the project that the student joins as part of their professional internship, selected in accordance with the elective study area (module) chosen by the student. Understanding the company's workflow, business processes, work-related risks, and participating in design, documentation development, or quality control, in alignment with the work process and the possibilities of the work environment.		
Course outcomes	Enhancing the student's ability to integrate into the workforce upon completing their studies. Developing responsibility, a professional approach to work, and teamwork communication skills. Supplementing the theoretical knowledge acquired in the academic program with practical insight into the issues studied within the student's field. Utilizing the expertise of professionals employed at the institution where the internship is conducted to broaden practical knowledge and motivate students. Gaining a clear understanding of how the acquired knowledge and skills from the academic program can be applied in practice.		
Course outline			
Theoretical teaching			
Practical teaching (exercises, OFE, study and research work)	The content of the professional practice is fully aligned with its objectives. The student becomes familiar with the company's structure and business goals, adapts their involvement to the chosen field of study, and duly fulfills work responsibilities in accordance with the duties of employees in the company. The student describes their own engagement during the internship and provides a critical reflection on their experience, knowledge, and skills acquired throughout the practice.		
Textbooks/references			
	1		
	2		
	3		
	4		
	5		
Number of classes of active education per week during semester/trimester/year			
Lectures	Exercises	OFE	Study and research work
			Other classes
			6
Teaching methods	The student generally independently selects a company from the public, private, or state sector in which they will complete their professional internship. The internship can also be carried out abroad, in which case the student additionally improves their foreign language skills. Based on the student's proposal, the head of the selected field/module approves the internship at the desired institution and, upon request, issues a written referral for the internship to the person responsible for conducting the practice at the given institution. Upon completion of the internship, and based on the student's report and confirmation from the responsible individual—verified with a signature and the company seal—the student is awarded 3 ECTS credits for the completed practice.		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises	70	Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program		Computer Science and Informatics	
Type and level of studies		Master Studies	
The name of the course		Master Thesis	
Lecturer (for lectures)			
Lecturer/associate (for exercises)			
Lecturer/associate (for OFE)			
Number of ECTS	9	Course status (obligatory/elective)	Obligatory
Prerequisites			
Course objectives	The Master's thesis aims to consolidate, confirm and practically apply the knowledge acquired during the Master's academic studies. The student is given the opportunity to demonstrate the ability to independently carry out a project, which can be of a practical, research or theoretical-methodological nature. The student also gains experience in presenting his/her work through a written form and an oral presentation during the thesis defense.		
Course outcomes	Ability to lead an independent project, ability to formulate and analyze problems, critically review possible solutions, review literature in a given field. Application of acquired engineering and design knowledge and skills to solve problems, taking into account the complexity, costs, reliability and efficiency of the solution. Ability to write a paper in a given form. Ability to clearly explain the completed project through an oral defense of the paper.		
Course outline			
Theoretical teaching	A master's thesis is an independent research, practical or theoretical methodological work of a student in accordance with the level of study, in which he becomes acquainted with a narrow field through a literature review and adopts the research methodology, or design, necessary for the preparation of the thesis. Through the preparation of the thesis, the student applies practical and theoretical knowledge acquired during his studies. The written thesis usually contains an introductory chapter, a definition of the problem, a review of the field and existing solutions, a proposal and description of the solution, a conclusion and literature. A public oral defense of the thesis is organized before a committee of three members, one of whom is the thesis mentor. During the oral defense, the candidate explains the results of his work, and then answers questions from the committee members, thereby demonstrating the candidate's ability to orally present his project.		
Practical teaching (exercises, OFE, study and research work)			
Textbooks/references			
1			
2			
3			
4			
5			
Number of classes of active education per week during semester/trimester/year			
Lectures	Exercises	OFE	Study and research work
			2
Teaching methods			
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	70
Exercises		Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Intelligent Systems		
Lecturer (for lectures)		Stoimenov V. Leonid, Frtunić-Gligorijević B. Milena		
Lecturer/associate (for exercises)		Frtunić-Gligorijević B. Milena		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	To provide students with insight into advanced artificial intelligence techniques, to present current problems and possible solutions for the implementation of intelligent systems. To introduce students to the importance and process of prototype realization, i.e. MVP (Minimum Valuable Product). Theoretical and practical knowledge of various areas of AI: the importance of computer vision, communication and planning for the implementation of intelligent systems, problems of inference with unreliable knowledge, elements of natural language processing, speech processing and synthesis, machine learning and neural networks. To present students with possible applications of intelligent systems for business applications in various real-world application domains. Presentation of existing libraries with implemented AI algorithms and possibilities of their application. Presentation of existing data sources.			
Course outcomes	At the end of the course, the student will be able to understand current problems in the implementation of intelligent systems, as well as future directions of research and development in artificial intelligence. The student will be able to respond to the challenges of selecting and designing individual parts of an intelligent system. The student will be able to recognize the problems of implementing distributed intelligent systems, the problem of semantic integration of information, and to implement some solutions using open source libraries or independently in the form of MVP.			
Course outline				
Theoretical teaching	Software engineering and artificial intelligence. The concept of MVP (Minimum Valuable Product) and how to apply it to systems based on artificial intelligence. The process of developing intelligent systems. Artificial intelligence techniques and their possible areas/domains of application. Computer vision. Communication: natural language processing. Speech recognition. Natural language understanding. Speech synthesis. Planning and planning algorithms. Probabilistic inference. Machine learning and machine learning algorithms. Neural networks and biologically inspired algorithms. Distributed intelligence and distributed inference systems. Application of intelligent systems in different domains. Examples of good practice.			
Practical teaching (exercises, OFE, study and research work)	Principles of software engineering for the development of intelligent systems. Implementation of systems in the form of an MVP (Minimum Viable Product). Introduction to algorithms and methods for computer vision. Algorithms and methods for natural language processing. Algorithms for speech recognition and synthesis. Machine learning algorithms and neural networks. Decision trees and business intelligence. Ontologies and semantic representation. Implementation and possible applications of intelligent systems. Examples of open-source libraries and their capabilities for implementing intelligent systems. Examples of data sources that can be used for the development of intelligent systems.			
Textbooks/references				
	1	S. Russell, P. Norvig: Artificial Intelligence: A Modern Approach, 4th edition, Pearson, 2020. ISBN 978-0134610993.		
	2	A. Biswas, W. Talukdar, Building Agentic AI Systems: Create intelligent, autonomous AI agents that can reason, plan, and adapt, Packt Publishing, 2025, ISBN-10 : 1803238755		
	3	Handbook on Artificial Intelligence-Empowered Applied Software Engineering, Eds. Maria Virvou, George A. Tsihrintzis, Nikolaos G. Bourbakis, Lakhmi C. Jain, Springer Cham, 2023, ISBN 978-3-031-08204-7		
	4	L. Bass, Q. Lu, I. Weber, L. Zhu, Engineering AI Systems: Architecture and DevOps Essentials, 2025, Addison-Wesley Professional, ISBN: 9780138261542		
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0

Teaching methods	Lectures, auditory exercises, laboratory exercises, independent work of students on homework and projects		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises	40	Oral exam	30
Colloquia			
Projects	30		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Data Mining Techniques and Methods		
Lecturer (for lectures)		Stojković R. Suzana		
Lecturer/associate (for exercises)		Marković M. Ivica		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The goal of this course is to introduce students to major data mining tasks and with special emphasis on the using the data mining techniques and methods into text analysis and information retrieval systems.			
Course outcomes	After completing this course, students should acquire theoretical knowledge of the principles of the work of the data mining tools and to be able to use existing data mining open source tools and to develop new.			
Course outline				
Theoretical teaching	Data mining goal and application. Types of datasets and attributes. Data quality problems. Data preprocessing. Similarity and desimilarity scores. Exploratory data analysis, summary statistics, data visualization. Classification of data mining algorithms. Linear regression. Classification algorithms: Naive Bayes algorithm, logistic regresion, SVM. Classifier evaluation. Clustering algorrithms: K-means, hierarchical clausterization. Time series. Characteristics of time series: stationarity, trend, seasonality, cyclicity. Linear processes: autoregressive (AR), moving average (MA) processes, and ARMA. Non-stationary time series. Methods for time series modeling, the Box-Jenkins method. Associative analysis. Algorithms for associative rules generation.			
Practical teaching (exercises, OFE, study and research work)	Implementation of all steps of data analysis using open-source libraries: Data preprocessing: outlier detection and removal, handling of missing values, normalization, standardization, dimensionality reduction. Data visualization. Linear regression. Classification algorithms. Evaluation of classification performance. Clustering algorithms. Time series analysis: time series preprocessing, stationarity tests, calculation of correlation and autocorrelation, ARIMA model. Creation of association rules.			
Textbooks/references				
1	Pang-Ning Tan, Anuj Karpatne, Michael Steinbach, Vipin Kumar: Introduction to data mining, 2nd Edition, Addison Wesley, 2019.			
2	D. T. Larose: "Data mining methods and models", JONN WILEY & SONS, 2006			
3	George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel, Greta M. Ljung: Time Series Analysis: Forecasting and Constrol, 5th Edition, Wiley, 2015			
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, practical demonstrations, consultations, seminars and projects presentation			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		20	Oral exam	40
Colloquia				
Projects		40		

Specification for the book of courses

Study program	Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering
Type and level of studies	Master Academic Studies
The name of the course	Big Data Systems
Lecturer (for lectures)	Stojanović H. Dragan, Stojanović M. Natalija
Lecturer/associate (for exercises)	Stojanović M. Natalija
Lecturer/associate (for OFE)	
Number of ECTS	6
	Course status (obligatory/elective) Elective
Prerequisites	Databases, Distributed systems
Course objectives	Mastering the knowledge, methods, and technologies necessary for the development of systems and applications for processing and analyzing large volumes of data (Big Data).
Course outcomes	Upon successful completion of the course, the student understands the fundamental concepts, architectures, and technologies for collecting, storing, processing, and analyzing large volumes of data. The student is able to design and implement distributed systems for Big Data processing using appropriate databases, distributed storage systems, and data processing platforms. The student is capable of applying technologies for processing stored data, large-scale data streams, and complex events in real or near-real time. The student is able to develop and apply Big Data systems in cloud, fog, and edge environments, as well as visualize and interpret analytical results. Through practical work, the student is able to independently analyze, implement, evaluate, and optimize Big Data applications in various real-world domains.
Course outline	
Theoretical teaching	Introduction to Big Data Processing and Analytics Systems. Characteristics, types, and formats of big data. Architecture and core components of Big Data systems. Methods and technologies for collecting and storing big data. NoSQL/NewSQL distributed databases (Cassandra, MongoDB, Redis, Druid, Spanner, etc.). Distributed file systems (GFS, HDFS), data lakes, and data lakehouse architectures. Methods and technologies for ingesting and streaming big data (Kafka, Pulsar, NiFi, Flume, etc.). Distributed systems and platforms for managing, processing, and analyzing large stored data (Hadoop, Spark, Flink, etc.). Systems for processing and analyzing fast and high-volume data streams and complex events (Spark Streaming, Flink, etc.). Visualization and visual analytics of big data (InfluxDB, Grafana, etc.). Big data management and analysis in distributed infrastructures across the cloud–fog–edge computing continuum. Analysis, design, implementation, and evaluation of systems and applications for big data processing and analytics. Modern applications based on big data processing and analysis in smart cities, healthcare, industry, transportation, agriculture, traffic systems, as well as social networks/media, and more.
Practical teaching (exercises, OFE, study and research work)	amiliarization with the characteristics, types, and formats of big data and conversion between them. Design and architecture of systems for big data processing and analytics, and implementation technologies. Implementation of methods and technologies for collecting and storing big data. Practical work on storing big data in NoSQL/NewSQL distributed databases: Cassandra, MongoDB, Redis, Druid, Spanner, etc. Storing big data in distributed file systems (GFS, HDFS), data lakes, and data lakehouse architectures. Application and implementation of methods and technologies for big data ingestion and transmission: Kafka, Pulsar, NiFi, etc. Implementation of a software system for big data processing on the Hadoop platform. Implementation of a software system for managing, processing, and analyzing stored big data using the Spark platform and relevant software libraries. Implementation of a distributed system for processing and analyzing big data streams using Spark Streaming. Implementation of a distributed system for processing and analyzing fast and large data streams and complex events using the Flink platform. Implementation of visualization of big data processing and analysis results using InfluxDB, Grafana, and Python libraries. Implementation of a distributed system for big data analysis within the distributed infrastructure of the computing continuum. Implementation of applications based on big data processing and analysis in smart domains: smart cities, healthcare, industry, transportation, agriculture, traffic systems, as well as social networks/media, etc., using publicly available or simulated datasets.
Textbooks/references	

	1	Nathan Marz, James Warren: Big Data Principles and best practices of scalable realtime data systems. Manning Publications Co., 2015.			
	2	Martin Kleppmann, Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems, O'Reilly Media, 2017.			
	3	Jules S. Damji, Brooke Wenig, Tathagata Das, and Denny Lee, Learning Spark: Lightning-Fast Data Analytics, 2nd edition, O'Reilly Media, 2020			
	4	Fabian Hueske and Vasiliki Kalavri, Stream Processing with Apache Flink, O'Reilly Media, 2019.			
	5	Adi Polak, Scaling Machine Learning with Spark: Distributed ML with MLlib, TensorFlow, and PyTorch, O'Reilly Media, 2023			
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods	Lectures, exercises, independent student work on assignments and projects, student seminars.				
Grade (maximum number of points 100)					
Pre-exam duties		Points	Final exam		Points
Activity during lectures			Written exam		
Exercises			Oral exam		40
Colloquia		40			
Projects		20			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Web Mining		
Lecturer (for lectures)		Stojković R. Suzana, Bogdanović D. Miloš		
Lecturer/associate (for exercises)		Bogdanović D. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Web technologies basics, Data structures and algorithms, Databases, Probability and statistics			
Course objectives	Within this course, students will learn the specifics of web document analysis and unstructured data analysis. They should also become familiar with the specific possibilities of applying web mining technologies.			
Course outcomes	Students will acquire theoretical and practical knowledge in the field of document content analysis, classification and clustering of web documents. They should also learn basic techniques and tools for web structure analysis and web access analysis.			
Course outline				
Theoretical teaching	Introduction to Web mining. Web document specifics. Techniques of Web documents content analysis. Document topic identification and ranking of terms in a document. Classification and clustering of documents. Sentiment analysis. Analysis of web structure, ranking documents by importance. Techniques for improving website ratings. Web access analysis. Discovery of patterns of website user behavior. Recommendation systems. Web visualization.			
Practical teaching (exercises, OFE, study and research work)	Introduction to basic Web mining tools and their application. Document classification algorithms, methodologies and tools. Log file structure and log analysis algorithms and tools. Website rating tools. Libraries for implementing machine learning algorithms used in Web mining. Libraries and tools for data visualization. Model training environments used in Web mining.			
Textbooks/references				
	1	Bing Liu, Web Data Mining-Exploring Hyperlinks, Contents, and Usage Data, Second Edition, July 2011, Springer		
	2	Wouter de Nooy, Andrej Mrvar, Vladimir Batagelj, Exploratory Social Network Analysis with Pajek, Cambridge University Press, 2011		
	3	Ulrich Matter, An Introduction to Web Mining: with Applications in R, Springer, 2025, ISBN 978-3-031-96637-8		
	4	Julian McAuley, Personalized Machine Learning, Cambridge University Press, 2022, ISBN 9781009003971		
	5	William L. Hamilton, Graph Representation Learning, Morgan & Claypool/Springer, 2020.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures and demonstration exercises using presentations and interactive work on the computer. Seminar papers and project.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	40
Colloquia				
Projects		60		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Pattern Recognition		
Lecturer (for lectures)		Vučković V. Vladan, Radmanović M. Miloš		
Lecturer/associate (for exercises)		Vučković V. Vladan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		The aim of the course is to introduce students to basic methods and techniques for pattern recognition and to acquire sufficient knowledge for independent research work in this field and the practical implementation of pattern recognition applications.		
Course outcomes		In this course, students will acquire knowledge about methods and techniques in the field of pattern recognition that will enable them to successfully solve problems and conduct independent research in the field of pattern recognition.		
Course outline				
Theoretical teaching		Pattern recognition systems, registration and measurement of object characteristics. Data acquisition, preprocessing and signal representation. Complexity reduction and classification. Statistical methods for pattern recognition. Classification based on minimum error. Classification based on maximum similarity. EM-algorithm. SVM (Support Vector Machine) classifiers. Stochastic finite state machines and discrete Markov models. Hidden Markov models. Real-time visual recognition systems. Systems for processing and recognizing digital images and scanned documents. OCR systems. Examples of pattern recognition applications.		
Practical teaching (exercises, OFE, study and research work)		Study research work: Preparation of seminar papers in the field of pattern recognition and oral presentation and defense of papers.		
Textbooks/references				
1		S. Pal, A. Pal, Pattern Recognition From Classical to Modern Approaches, World Scientific, 2001.		
2		S. Bow, Pattern Recognition and Image Preprocessing, CRC Press, 2002.		
3		R. Duda, P. Hart, D. Stork, Pattern Classification, Wiley, 2000.		
4		Teaching material on the website: http://cs.elfak.ni.ac.rs/nastava/		
5		D. Dodou, Pattern Recognition and Classification: An Introduction, 2020, 2nd ed. Delft University of Technology Press.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures and demonstration exercises using slides and demo examples. Independent research work by students, preparation, presentation and defense of seminar papers. Implementation of projects.		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		20	Oral exam	50
Colloquia				
Projects		30		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Data Visualization		
Lecturer (for lectures)		Janković S. Dragan		
Lecturer/associate (for exercises)		Rajković J. Petar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Introducing students to the principles underlying data and information visualization in accordance with the purpose and field. Enabling students to use different methods for data visualization. Introducing existing tools and environments for visualizing data.			
Course outcomes	Students should be introduced to data representation methods and areas of their application. Upon completion of the course, students should be able to select an adequate visual representation for a specific problem, which they can implement in one of the existing tools or develop their own display.			
Course outline				
Theoretical teaching	Data types. Principles of data visualization. Perception. Optical illusions. Tables, charts, trees, graphs, treemaps. Multivariate data visualization. Animations, multimedia presentations. Social network visualization. Medical data visualization, Economic data visualization. Time series. GIS data visualization. Process visualization. Visualization tools and environments (Excel, Tableau, R). Data visualization and analysis. BI environments.			
Practical teaching (exercises, OFE, study and research work)	Practical teaching of visualization of various types of data using: Excel, Tableau, D3.js. Integration of visual data displays into software solutions. Software implementation of individual data visualization algorithms.			
Textbooks/references				
1	Tamara Munzner, Visualization Analysis and Design, CRC Press, 2014.			
2	Scott Murray, "Interactive Data Visualization for the Web", O'Reilly Media, 2013			
3	Claus O. Wilke, "Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures", O'Reilly Media, 2019.			
4	Jonathna Schwabish, "Better Data Visualizations: A Guide for Scholars, Researchers, and Wonks", Columbia University Press, 2021.			
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, Exercises, Laboratory Exercises			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		20	Oral exam	40
Colloquia				
Projects		40		

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Methods of optimization		
Lecturer (for lectures)			Marinković D. Slađana, Džunić S. Jovana		
Lecturer/associate (for exercises)			Damnjanović G. Ivan		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Obligatory / Elective
Prerequisites	None				
Course objectives	Mastering basic theoretical and practical knowledge in the field of optimization and getting acquainted with various optimization methods. Developing skills of mathematical modelling of real problems of practice, as well as solving them.				
Course outcomes	Students' competence to apply acquired knowledge in the profession. Capability to recognize optimization problems from practice, formulate them precisely, form mathematical models, select appropriate methods for their solution and apply the methods.				
Course outline					
Theoretical teaching	Real multivariable functions. Partial derivatives, gradient, Hessian. Local, constrained and global extreme values. Taylor's formula. Elements of convex analysis. General optimization problem. Linear programming. Duality principle in linear programming. Simplex method. Nonlinear programming. Overview of methods for unconstrained optimization (search methods, gradient methods). Constrained optimizations: necessary and sufficient conditions for optimality. Penalty function methods.				
Practical teaching (exercises, OFE, study and research work)	Solving problems that follow the content presented in the theoretical classes. Special emphasis is placed on looking at the problem as a whole, from recognizing the subject of optimization, through forming a mathematical model and choosing an appropriate method to solving and interpreting the results. Implementation of optimization algorithms through program packages for symbolic calculation. Through the preparation of homework assignments and the analysis of their solutions, the application of mathematical and software tools in the context of optimization is gradually mastered.				
Textbooks/references					
1	E.K.P. Chong, S.H. Zak, An Introduction to Optimization, John Wiley & Sons, 2001.				
2	X.S. Yang, Introduction to Mathematical Optimization, Cambridge International Science Publishing, 2008.				
3	M.J. Kochenderfer, T.A. Wheeler, Algorithms for Optimization, second edition, The MIT Press, 2025.				
4					
5					
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods	Lectures, computational exercises, computer exercises, project.				
Grade (maximum number of points 100)					
Pre-exam duties		Points	Final exam		Points
Activity during lectures			Written exam		40
Exercises		10	Oral exam		10
Colloquia					
Projects		40			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Machine Learning		
Lecturer (for lectures)		Stoimenov V. Leonid, Stanimirović S. Aleksandar		
Lecturer/associate (for exercises)		Stanimirović S. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Provide students with knowledge about machine learning, its significance, and its role in the development of intelligent systems for analyzing data from various domains in engineering and science. Enable students to gain insight into the fundamental methods of modern machine learning. Provide information on the ways to implement and apply individual machine learning techniques.		
Course outcomes		Insight into the fundamental methods of modern machine learning. Acquiring knowledge about the role of machine learning in the development of intelligent systems. Theoretical knowledge of the basics of machine learning, including the most important algorithms for supervised and unsupervised learning. Practical application and implementation of machine learning algorithms and/or the use of existing open-source libraries.		
Course outline				
Theoretical teaching		Introduction to machine learning. Approaches and types of machine learning: supervised, unsupervised, and reinforcement learning. The role of machine learning in intelligent systems and big data analysis. Data preparation for machine learning, emphasizing the importance of data quality. Descriptive data analysis. Supervised learning: algorithms and applications in classification and regression. Evaluation and selection of machine learning models. Methods for assessing model performance and criteria for selection. Dimensionality reduction. Techniques for feature reduction, attribute selection, and visualization. Unsupervised learning: discovering patterns in unlabeled data. Reinforcement learning: basic concepts and applications. Artificial neural networks: structure, learning, and applications.		
Practical teaching (exercises, OFE, study and research work)		Practical instruction includes independent or team work by students on identifying real-world problems that can be solved using machine learning techniques. Students design, develop, and evaluate solutions using modern programming languages and tools (Python, Jupyter, Scikit-learn, PyTorch, TensorFlow, Pandas, NumPy, etc.). The focus is on the entire problem-solving process: from data collection and preparation, through selection and application of appropriate algorithms, to model evaluation and result visualization. Practical instruction places special emphasis on: formulating problems from real-world or academic contexts, reasoned algorithm selection, critical analysis of obtained results, and collaborative work and presentation of solutions.		
Textbooks/references				
1	S. Russel, P. Norvig, Artificial intelligence - A Modern Approach, Pearson, 3rd edition, (2016), ISBN-10: 1292153962, ISBN-13: 978-1292153964			
2	G. Hulten, Building Intelligent Systems: A Guide to Machine Learning Engineering, 1st ed. edition (2018), ISBN-10: 1484234316, ISBN-13: 978-1484234310			
3	C. Sammut (Editor), G. I. Webb (Editor), Encyclopedia of Machine Learning and Data Mining, Springer, 2nd ed. (2017), ISBN-10: 148997685X, ISBN-13: 978-1489976857			
4	C.Albon, Machine Learning with Python Cookbook: Practical Solutions from Preprocessing to Deep Learning, O'Reilly Media, (2018), ISBN-13 : 978-1491989388			
5	J. Brownlee, Statistical Methods for Machine Learning: Discover How to Transform Data into Knowledge with Python, Machine Learning Mastery, (2018)			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, whiteboard exercises, lab exercises, independent student work on homework assignments and projects, student seminars (presentations of student work followed by discussions).			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		20	Written exam	
Exercises		20	Oral exam	30

Colloquia			
Projects	30		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Natural Language Processing		
Lecturer (for lectures)		Stojković R. Suzana		
Lecturer/associate (for exercises)		Đorđević D. Nikola		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The goal of this course is to introduce students to the basic concepts and ideas of the Natural Language Processing (NLP) and with applications of these concepts in information extraction, information retrieval systems, sentiment analysis, question answering systems, and text summarization. Special emphasis is placed on using large language models in these applications.			
Course outcomes	After completing this course the student acquires theoretical and practical knowledge necessary for development of applications based on natural language processing.			
Course outline				
Theoretical teaching	Introduction to NLP. Text tokenizing. Word normalization: stemming and lematization. Type errors correcting, minimum edit distance algorithm. Sentence segmentation. Statistical language models. N-gram model. Part-of-speech tagging. Syntax analysis. CKY algorithm. Semantic analysis. Syntax-directed semantic analysis. Semantic grammars. Text vectorization: TF-IDF weight vectors, word embeddings, sentence embeddings. Large language models. Text classification. Information extraction. Sentiment analysis. Question answering systems, RAG systems. Text summarization.			
Practical teaching (exercises, OFE, study and research work)	Text processing using classical algorithms and open-source libraries such as NLTK: Tokenization. Sentence segmentation. Part-of-speech tagging. Phrase extraction. Stemming and lemmatization. Named entity recognition. Text classification. Sentiment analysis. Text processing using fine-tuning of large language models: Named entity recognition. Text classification. Sentiment analysis. Summarization. Implementation of RAG systems.			
Textbooks/references				
1	D. Jurafsky and J. H. Martin: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models, Third Edition, McGraw Hill, 2025.			
2	S. Bird, E. Klein, E.: Natural Language Processing With Python, O'REILLI 2009.			
3				
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, practical demonstrations, consultations, seminars and projects presentation			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		20	Oral exam	40
Colloquia				
Projects		40		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Social Networks Analysis		
Lecturer (for lectures)		Tošić B. Milorad		
Lecturer/associate (for exercises)		Nejković M. Valentina		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Information Systems, Algorithms, Web Programming, Graph Theory, and Artificial Intelligence.			
Course objectives	Acquisition of practical programming skills, theoretical knowledge, and a systematic approach necessary for the design, implementation, and operation of systems in which information technologies, computers, the Internet, and people actively participate in system development.			
Course outcomes	Students are able to model, analyze, and visualize social networks using graph-theoretic methods, ranking algorithms, similarity analysis, and community detection. They are capable of extracting and analyzing social network data and applying machine learning methods for classification, filtering, and recommendation generation. They possess practical skills for implementing and evaluating algorithms on real-world social network data.			
Course outline				
Theoretical teaching	General conceptual assumptions: mathematical foundations, fundamental technologies. Algorithms based on HITS and PageRank methods. Similarity analysis and community detection in social network structures. Information extraction and intelligent reasoning from social networks. Systems for data collection from social media (tag extraction, content extraction, blog search, intelligent web crawling). Intelligent search. Recommendation generation, filtering, and ranking. Feature identification. Advanced classification on social networks. Distributed knowledge in systems with network structure. Visualization of social networks and data analysis. Practical applications and case studies in social network analysis.			
Practical teaching (exercises, OFE, study and research work)	Introduction to the mathematical foundations of social networks – graphs, nodes, edges, adjacency and incidence matrices. Fundamental technologies for social network analysis – data import and processing using tools such as Python (NetworkX) and Gephi. Implementation of the HITS algorithm – calculating authorities and hubs on small network examples. Implementation of the PageRank algorithm – application to web graphs and analysis of ranking convergence. Node similarity analysis – similarity measures and their application in community detection. Community detection in social networks – clustering algorithms. Information extraction from social networks – tag and content analysis using NLP tools. Intelligent search – implementation and analysis of search algorithms optimized for social networks. Recommendation generation – collaborative and content-based filtering, ranking of recommendations. Feature identification and advanced classification – applying machine learning to analyze user behavior. Social network visualization – tools and techniques for visualizing complex network structures. Practical applications and case studies – analysis of real social network data and report preparation.			
Textbooks/references				
	1	Yang, Song, ed. Social Network Analysis in Action: Basic Methods and Applications. Springer, 2025.□		
	2	Zhang, Yanchun, Jeffrey Xu Yu, and Jingyu Hou. Web communities analysis and construction. SpringerVerlag, 2012		
	3	McLevey, John, John Scott, and Peter J. Carrington, eds. The SAGE Handbook of Social Network Analysis. 2nd ed. SAGE, 2023.		
	4	Alag, Satnam. Collective intelligence in action. Manning, 2009.		
	5	Galety, Mohammad Gouse, Chiai Al Atroshi, Buni Balabantaray, and Sachi Nandan Mohanty, eds. Social network analysis: theory and applications. John Wiley & Sons, 2022.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0

Teaching methods	Lectures; Classroom exercises; Laboratory exercises; Computer exercises; Consultations; Independent student research; Oral presentations by students on chosen/assigned topics; Active student participation in classes through the use of the interactive course website.		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures	30	Written exam	
Exercises	30	Oral exam	40
Colloquia			
Projects			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Deep Learning		
Lecturer (for lectures)		Milosavljević Lj. Aleksandar		
Lecturer/associate (for exercises)		Milosavljević Lj. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Basic knowledge of Python programming, linear algebra, and machine learning fundamentals.			
Course objectives	Introducing students to the field of deep learning. Introduction to basic concepts, training techniques, and architectures of deep neural networks.			
Course outcomes	Knowledge of the basic architectures, techniques, and algorithms used in training deep neural networks. Understanding the advantages of deep learning over traditional approaches. Knowledge of domains where deep learning techniques are applicable and yield good results.			
Course outline				
Theoretical teaching	Introduction to the field of deep learning and historical context. Artificial neuron and artificial neural networks. Neural networks training, error function and parameter optimization. Backpropagation algorithm. Convolutional neural networks. Activation functions, parameter initialization, rejection, batch normalization. Parameter updating during training, network ensembles and voting, data augmentation, transfer learning. Deep learning hardware and software. Convolutional neural network architectures. Visualization and understanding of deep neural networks. Recurrent neural networks. Semantic segmentation, object detection, instance extraction. Generative models, autoencoders, generative adversarial networks. Transformer architecture and large language models. Deep reinforcement learning.			
Practical teaching (exercises, OFE, study and research work)	Introduction to the Python programming language, TensorFlow library, and Keras API. Comparative implementation of training a two-layer neural network using Numpy, basic TensorFlow, and Keras API. Handwritten digit recognition (MNIST dataset) using a layered neural network and a convolutional neural network. Image classification (Caltech-101 dataset) using a pretrained convolutional network and transfer learning. Visualization of the operation of a convolutional neural network for image classification using the Grad-CAM algorithm. Semantic image segmentation (PASCAL VOC dataset) using a U-Net network. Image object detection using a YOLO network. Text sentiment analysis (IMDB dataset) using recurrent neural networks. Text machine translation using an LSTM network. Implementation of a variational autoencoder using the MNIST dataset. Additional training of the transformer network (GPT-2) for text generation using "Serbian folk songs".			
Textbooks/references				
1	Francois Chollet, Deep Learning with Python, 3rd Edition, Manning, 2025.			
2	Howard Huang, Eli Stevens, Luca Antiga, Thomas Viehmann, Deep Learning with PyTorch, 2nd Edition, Manning, 2026.			
3	Ian Goodfellow, Yoshua Bengio, Aaron Courville, Deep Learning, MIT Press, 2016.			
4	Material from the website http://cs231n.stanford.edu/			
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, auditory exercises, independent work of students on projects.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	30	Oral exam		40
Colloquia				
Projects	30			

Specification for the book of courses

Study program		Computer Science and Informatics		
Type and level of studies		Master Studies		
The name of the course		Master Thesis – Study and Research Work		
Lecturer (for lectures)				
Lecturer/associate (for exercises)				
Lecturer/associate (for OFE)				
Number of ECTS	6	Course status (obligatory/elective)	Obligatory	
Prerequisites				
Course objectives	Application of basic, theoretical-methodological, scientific-professional and professional-applied knowledge and methods to solve specific problems. The student studies the problem, its structure and complexity and, based on the conducted analyses, draws conclusions about possible ways of solving it. By studying the literature, the student becomes familiar with the methods intended for solving similar tasks and engineering practice in solving them.			
Course outcomes	Enabling students to independently apply previously acquired knowledge from various areas they have studied, in order to observe the structure of a given problem and its systemic analysis in order to draw conclusions about possible directions for its solution. Through the independent use of literature, students expand their knowledge by studying various methods and works related to similar issues. In this way, students develop the ability to conduct analyses and identify problems within the given issue. Through the practical application of acquired knowledge, students develop the ability to perceive the place and role of engineers in the chosen area, the need for cooperation with other professions and teamwork.			
Course outline				
Theoretical teaching	It is formed individually in accordance with the needs of a specific diploma-master's thesis, its complexity and structure. The student, according to his affinities and preferences, chooses the field of study work, or rather the subject teacher from the list of teachers in the study program who defines a specific task for him. The student studies professional literature, professional and scientific works that deal with similar topics, performs analyses in order to find a solution to a specific task or performs certain experiments in the laboratory. The study work also includes active monitoring of primary knowledge, organization and performance of experiments, numerical simulations and statistical data processing, and preparation of a seminar paper in a narrow scientific and educational field to which the topic of the independent research work belongs.			
Practical teaching (exercises, OFE, study and research work)				
Textbooks/references				
1				
2				
3				
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
			12	
Teaching methods				
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	50
Colloquia				
Projects		50		

Specification for the book of courses

Study program		Computer Science and Informatics	
Type and level of studies		Master Studies	
The name of the course		Professional Practice	
Lecturer (for lectures)			
Lecturer/associate (for exercises)			
Lecturer/associate (for OFE)			
Number of ECTS	3	Course status (obligatory/elective)	Obligatory
Prerequisites			
Course objectives	Familiarization with the work process in the company where the professional internship is conducted, its objectives, and organizational units. Getting to know the team and the project that the student joins as part of their professional internship, selected in accordance with the elective study area (module) chosen by the student. Understanding the company's workflow, business processes, work-related risks, and participating in design, documentation development, or quality control, in alignment with the work process and the possibilities of the work environment.		
Course outcomes	Enhancing the student's ability to integrate into the workforce upon completing their studies. Developing responsibility, a professional approach to work, and teamwork communication skills. Supplementing the theoretical knowledge acquired in the academic program with practical insight into the issues studied within the student's field. Utilizing the expertise of professionals employed at the institution where the internship is conducted to broaden practical knowledge and motivate students. Gaining a clear understanding of how the acquired knowledge and skills from the academic program can be applied in practice.		
Course outline			
Theoretical teaching			
Practical teaching (exercises, OFE, study and research work)	The content of the professional practice is fully aligned with its objectives. The student becomes familiar with the company's structure and business goals, adapts their involvement to the chosen field of study, and duly fulfills work responsibilities in accordance with the duties of employees in the company. The student describes their own engagement during the internship and provides a critical reflection on their experience, knowledge, and skills acquired throughout the practice.		
Textbooks/references			
	1		
	2		
	3		
	4		
	5		
Number of classes of active education per week during semester/trimester/year			
Lectures	Exercises	OFE	Study and research work
			Other classes
			6
Teaching methods	The student generally independently selects a company from the public, private, or state sector in which they will complete their professional internship. The internship can also be carried out abroad, in which case the student additionally improves their foreign language skills. Based on the student's proposal, the head of the selected field/module approves the internship at the desired institution and, upon request, issues a written referral for the internship to the person responsible for conducting the practice at the given institution. Upon completion of the internship, and based on the student's report and confirmation from the responsible individual—verified with a signature and the company seal—the student is awarded 3 ECTS credits for the completed practice.		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises	70	Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program		Computer Science and Informatics	
Type and level of studies		Master Studies	
The name of the course		Master Thesis	
Lecturer (for lectures)			
Lecturer/associate (for exercises)			
Lecturer/associate (for OFE)			
Number of ECTS	9	Course status (obligatory/elective)	Obligatory
Prerequisites			
Course objectives	The Master's thesis aims to consolidate, confirm and practically apply the knowledge acquired during the Master's academic studies. The student is given the opportunity to demonstrate the ability to independently carry out a project, which can be of a practical, research or theoretical-methodological nature. The student also gains experience in presenting his/her work through a written form and an oral presentation during the thesis defense.		
Course outcomes	Ability to lead an independent project, ability to formulate and analyze problems, critically review possible solutions, review literature in a given field. Application of acquired engineering and design knowledge and skills to solve problems, taking into account the complexity, costs, reliability and efficiency of the solution. Ability to write a paper in a given form. Ability to clearly explain the completed project through an oral defense of the paper.		
Course outline			
Theoretical teaching	A master's thesis is an independent research, practical or theoretical methodological work of a student in accordance with the level of study, in which he becomes acquainted with a narrow field through a literature review and adopts the research methodology, or design, necessary for the preparation of the thesis. Through the preparation of the thesis, the student applies practical and theoretical knowledge acquired during his studies. The written thesis usually contains an introductory chapter, a definition of the problem, a review of the field and existing solutions, a proposal and description of the solution, a conclusion and literature. A public oral defense of the thesis is organized before a committee of three members, one of whom is the thesis mentor. During the oral defense, the candidate explains the results of his work, and then answers questions from the committee members, thereby demonstrating the candidate's ability to orally present his project.		
Practical teaching (exercises, OFE, study and research work)			
Textbooks/references			
	1		
	2		
	3		
	4		
	5		
Number of classes of active education per week during semester/trimester/year			
Lectures	Exercises	OFE	Study and research work
			2
Teaching methods			
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	70
Exercises		Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering	
Type and level of studies			Master Academic Studies	
The name of the course			Advanced Software Engineering	
Lecturer (for lectures)			Rančić D. Dejan, Milosavljević Lj. Aleksandar	
Lecturer/associate (for exercises)			Mihajlović T. Vladan	
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Basic knowledge of object-oriented programming, databases, and software engineering fundamentals.			
Course objectives	Mastering advanced methods for software development and evolution and methods for measuring the quality of software products and processes.			
Course outcomes	The student is equipped to apply agile methodologies in software planning and development. The student is capable of using advanced version control and project management systems. The student knows how to analyze metrics and apply techniques for continuous software quality improvement.			
Course outline				
Theoretical teaching	A brief overview and history of software engineering. Software evolution. Software metrics. Software cost estimation. Quality management. Software process improvement. Agile methods for software development. Introduction to Scrum. User stories. Agile planning and estimating. Configuration management. Code control systems. Software reuse. Component-based software engineering. Prototyping. Software metrics. Development of safety-sensitive software. Development of real-time software.			
Practical teaching (exercises, OFE, study and research work)	Introduction to the SVN and GIT code control systems and the basic operations they offer. Best practices in organizing branches in GIT code in software development. Solving code merging problems. Introduction to project management and bug tracking support systems Redmine and JIRA. Support for agile software development in JIRA. Introduction to current software development technologies through seminars.			
Textbooks/references				
1	Ian Sommerville, Engineering Software Products: An Introduction to Modern Software Engineering, Pearson, 2020.			
2	Roger Pressman, Bruce Maxim, Software Engineering: A Practitioner's Approach, 9th ed., McGraw-Hill, 2020.			
3	Robert C. Martin, Clean Agile: Back to Basics, Prentice Hall, 2019.			
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, auditory exercises, independent work of students on projects.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		30	Oral exam	40
Colloquia				
Projects		30		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Intelligent Systems		
Lecturer (for lectures)		Stoimenov V. Leonid, Frtunić-Gligorijević B. Milena		
Lecturer/associate (for exercises)		Frtunić-Gligorijević B. Milena		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		To provide students with insight into advanced artificial intelligence techniques, to present current problems and possible solutions for the implementation of intelligent systems. To introduce students to the importance and process of prototype realization, i.e. MVP (Minimum Valuable Product). Theoretical and practical knowledge of various areas of AI: the importance of computer vision, communication and planning for the implementation of intelligent systems, problems of inference with unreliable knowledge, elements of natural language processing, speech processing and synthesis, machine learning and neural networks. To present students with possible applications of intelligent systems for business applications in various real-world application domains. Presentation of existing libraries with implemented AI algorithms and possibilities of their application. Presentation of existing data sources.		
Course outcomes		At the end of the course, the student will be able to understand current problems in the implementation of intelligent systems, as well as future directions of research and development in artificial intelligence. The student will be able to respond to the challenges of selecting and designing individual parts of an intelligent system. The student will be able to recognize the problems of implementing distributed intelligent systems, the problem of semantic integration of information, and to implement some solutions using open source libraries or independently in the form of MVP.		
Course outline				
Theoretical teaching		Software engineering and artificial intelligence. The concept of MVP (Minimum Valuable Product) and how to apply it to systems based on artificial intelligence. The process of developing intelligent systems. Artificial intelligence techniques and their possible areas/domains of application. Computer vision. Communication: natural language processing. Speech recognition. Natural language understanding. Speech synthesis. Planning and planning algorithms. Probabilistic inference. Machine learning and machine learning algorithms. Neural networks and biologically inspired algorithms. Distributed intelligence and distributed inference systems. Application of intelligent systems in different domains. Examples of good practice.		
Practical teaching (exercises, OFE, study and research work)		Principles of software engineering for the development of intelligent systems. Implementation of systems in the form of an MVP (Minimum Viable Product). Introduction to algorithms and methods for computer vision. Algorithms and methods for natural language processing. Algorithms for speech recognition and synthesis. Machine learning algorithms and neural networks. Decision trees and business intelligence. Ontologies and semantic representation. Implementation and possible applications of intelligent systems. Examples of open-source libraries and their capabilities for implementing intelligent systems. Examples of data sources that can be used for the development of intelligent systems.		
Textbooks/references				
1		S. Russell, P. Norvig: Artificial Intelligence: A Modern Approach, 4th edition, Pearson, 2020. ISBN 978-0134610993.		
2		A. Biswas, W. Talukdar, Building Agentic AI Systems: Create intelligent, autonomous AI agents that can reason, plan, and adapt, Packt Publishing, 2025, ISBN-10 : 1803238755		
3		Handbook on Artificial Intelligence-Empowered Applied Software Engineering, Eds. Maria Virvou, George A. Tsihrintzis, Nikolaos G. Bourbakis, Lakhmi C. Jain, Springer Cham, 2023, ISBN 978-3-031-08204-7		
4		L. Bass, Q. Lu, I. Weber, L. Zhu, Engineering AI Systems: Architecture and DevOps Essentials, 2025, Addison-Wesley Professional, ISBN: 9780138261542		
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0

Teaching methods	Lectures, auditory exercises, laboratory exercises, independent work of students on homework and projects		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises	40	Oral exam	30
Colloquia			
Projects	30		

Specification for the book of courses

Study program	Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering Control Systems / Automatic Control
Type and level of studies	Master Academic Studies
The name of the course	Geospatial Information Systems
Lecturer (for lectures)	Stojanović H. Dragan, Predić B. Bratislav
Lecturer/associate (for exercises)	Predić B. Bratislav
Lecturer/associate (for OFE)	
Number of ECTS	6 Course status (obligatory/elective) Obligatory / Elective
Prerequisites	Databases, Information systems
Course objectives	Mastering the knowledge, methods, and technologies necessary for the design and implementation of geospatial information systems (GIS) and applications, as well as the application of spatial data science.
Course outcomes	Upon successful completion of the course, the student understands the fundamental concepts, methods, and technologies of geospatial information systems and spatial data science. The student is able to collect, model, store, process, and analyze raster, vector, and spatiotemporal geospatial data. The student is capable of designing and implementing geospatial databases, applying spatial indexing structures, and developing efficient geospatial queries and analyses. The student is able to apply standards and technologies for developing Web GIS, OGC services, mobile GIS applications, and location-based services. Through practical work, the student is able to use modern GIS tools, libraries, and platforms for visualization, spatial analysis, mobility analysis, and processing of large-scale geospatial data.
Course outline	
Theoretical teaching	Introduction to Geospatial Information Systems (GIS). Geospatial coordinate systems. Methods and systems for positioning and collecting geospatial data: GPS, remote sensing, aerial and orthophoto imaging, RADAR, LIDAR, conversion of existing data, OpenStreetMap (OSM), etc. Raster representation of geospatial data, formats, and processing functions. Vector representation of geospatial data, formats, functions, and algorithms for their processing and analysis. Geospatial databases. Indexing structures and methods for accessing geospatial data. Time in GIS and management of spatiotemporal data. Visualization of geospatial data, interaction methods, and visual analytics. Analysis of geospatial data and spatial data science. Analysis of mobile data, trajectories, spatial networks – mobility data science. Processing and analysis of large geospatial data sets. Specifications and standards in geospatial data and GIS (OGC, ISO TC 211). Web GIS and OGC-specified GI services (WMS, WFS, WCS, WPS, etc.). Mobile GIS and location-based services.
Practical teaching (exercises, OFE, study and research work)	Basic Functions and Components of GIS (QGIS, ArcGIS Pro). Data representation in different coordinate systems. Methods and technologies for managing geospatial data in GIS. Application of libraries and functions for processing and analyzing raster data (GDAL, rasterio). Application of libraries and functions for processing and analyzing vector data (GEOS, shapely). Design and implementation of geospatial databases (PostGIS/PostgreSQL). Application of indexing structures in processing geospatial queries (PostGIS). Processing and analysis of spatiotemporal data and GIS. Visualization and visual analysis of geospatial data (Leafmap, Folium, etc.). Geospatial data analysis and implementation of spatial data science (SAGA, GeoPandas, PySal). Mobility and trajectory data analysis (MovingPandas, scikit-mobility). Systems for processing and analyzing large geospatial data (GeoSpark, Sedona, Mosaic). Implementation and application of OGC Web GIS services (GeoServer, pygeoapi). Development of Web GIS applications using libraries and frameworks (Leaflet, OpenLayers). Development of mobile GIS applications and location-based services on Android/iOS platforms.
Textbooks/references	
1	Kang-tsung Chang, Introduction to Geographic Information Systems, 9th Edition, McGraw-Hill Science, 2019
2	Matt Duckham, Qian (Chayn) Sun, Michael F. Worboys, GIS: A Computing Perspective, 3rd edition, CRC Press, 2024.
3	Michael Dorman, Anita Graser, Jakub Nowosad, Robin Lovelace, Geocomputation with Python, Chapman and Hall/CRC, 2025.
4	Michael Dorman, Introduction to Web Mapping, Chapman and Hall/CRC, 2020.

5	Shashi Shekhar, Sanjay Chawla, Spatial Databases: A Tour, Pearson; 1st edition, 2003			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, exercises, independent student work on assignments and projects, student seminars.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	40
Colloquia		40		
Projects		20		

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Advanced Web Technologies		
Lecturer (for lectures)			Petković M. Ivan		
Lecturer/associate (for exercises)			Petković M. Ivan		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Obligatory / Elective
Prerequisites	None				
Course objectives	Introduction to advanced concepts of web application development, including the use of high-performance and memory-safe programming languages.				
Course outcomes	Students should acquire sufficient theoretical and practical knowledge to design and implement complex, high-performance, and secure web applications using the Rust programming language				
Course outline					
Theoretical teaching	High-performance languages. Classification of languages based on memory management. The problem of memory safety. Advantages and disadvantages of object-oriented and functional programming. Core concepts of Rust. Variable ownership and borrow checking. Mutable and immutable references. Structs. Enumerations. Handling null values. Returning results. Variable lifetimes. Inheritance and composition. Smart pointers. Lambda functions. Closures and functional programming. Asynchronous programming. Metaprogramming using macros. Rust and WebAssembly. Full stack Rust web applications.				
Practical teaching (exercises, OFE, study and research work)	Development tools and environment for Rust. Package manager - Cargo. Variables and references. Functions. Data structures and behaviors (traits). Enumerations. Option and null values. Result and function return handling. Introduction to Rust API implementation environment. Controllers. Validation using attributes. Database access via SQL mappers (sqlx, etc.). SQL Mappers or ORM. Database migrations. Query validation during compilation. Writing tests. Defining multiple environments (test, production). Exception and error handling. Best practices for writing Rust web applications.				
Textbooks/references					
1	Steve Klabnik & Carol Nichols, The Rust Programming Language, No Starch Press, 2019				
2	Jim Blandy, Jason Orendorff, Leonora F. Tindall, Programming Rust: Fast, Safe Systems Development (2nd Edition), O'Reilly Media, 2021				
3	Jon Gjengset, Rust for Rustaceans: Idiomatic Programming for Experienced Developers, No Starch Press, 2021				
4	Bastian Gruber, Rust Web Development, Packt Publishing, 2023				
5	Luca Palmeri, Zero To Production In Rust: An introduction to backend development, 2022				
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods	Lectures and computer-based demonstration exercises. Independent hands-on practice sessions.				
Grade (maximum number of points 100)					
Pre-exam duties		Points	Final exam		Points
Activity during lectures		5	Written exam		
Exercises		20	Oral exam		30
Colloquia					
Projects		45			

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Advanced Operating Systems		
Lecturer (for lectures)			Stanimirović S. Aleksandar, Stojanović H. Dragan		
Lecturer/associate (for exercises)			Stanimirović S. Aleksandar		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Elective
Prerequisites		Operating systems, Distributed systems			
Course objectives		Understanding the technologies, development trends, as well as the design and implementation of modern operating systems and system software development.			
Course outcomes		Upon successful completion of the course, the student understands advanced concepts, architectures, and principles of the design and implementation of modern operating systems and system software. The student is able to analyze and apply principles of kernel development and extension, as well as embedded and real-time operating systems in various computing and IoT environments. The student is capable of designing and implementing virtualized and containerized environments and applying mechanisms for container management, scheduling, and orchestration. The student is able to develop, monitor, and scale cloud-native applications using Docker and Kubernetes platforms and appropriate monitoring tools. Through practical work, the student is able to integrate modern system software, virtualization, containerization, and Infrastructure as Code technologies in the development of reliable and scalable computing systems.			
Course outline					
Theoretical teaching		Introduction to Advanced Operating Systems and System Software. Advanced concepts, design, and implementation of modern operating systems. Architecture, design, and implementation of Windows, Unix/Linux, and macOS. Development of the Linux kernel. Extending and rebuilding the Linux kernel and implementing kernel modules. Development of embedded Linux kernels. Real-time operating systems for microcontrollers. Development of IoT systems based on embedded Linux and real-time OS. Virtualization and virtual machines (VMware). Container-based virtualization (Docker). Deployment, scheduling, and orchestration of containers (Kubernetes). Development of cloud-native applications using Docker and Kubernetes. Monitoring and scaling execution of cloud-native applications. Infrastructure as Code (IaC).			
Practical teaching (exercises, OFE, study and research work)		Development of advanced concepts in operating systems and system software. Design and implementation of the Windows operating system. Implementation of Linux kernel components. Implementation of Linux kernel extensions and kernel modules. Development of embedded Linux kernel using the Yocto Project. Implementation of real-time operating systems (Zephyr, ROS). Development of IoT systems based on embedded Linux and real-time OS (Zephyr). Virtualization and virtual machines (VMware, ESXi, VirtualBox). Container-based virtualization using the Docker platform. Implementation of methods and techniques for container deployment, scheduling, and orchestration using the Kubernetes platform. Development of cloud-native applications using Docker and Kubernetes. Monitoring and scaling of cloud-native applications (Prometheus, Grafana). Infrastructure as Code (IaC) using Terraform.			
Textbooks/references					
	1	Andrew S. Tanenbaum, Herbert Bos, Modern Operating Systems, 5th edition, Pearson Education/Prentice-Hall, 2022			
	2	Robert Love, Linux Kernel Development (3rd Edition), Addison-Wesley Professional; 2010			
	3	Frank Vasquez, Chris Simmonds, Mastering Embedded Linux Development, 4th Edition, Packt Publishing, 2025.			
	4	Ashley Davis, Bootstrapping Microservices with Docker, Kubernetes, and Terraform, Manning Publications, 2021.			
	5				
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods	Lectures, exercises, independent student work on assignments and projects, student seminars.				
Grade (maximum number of points 100)					

Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises		Oral exam	40
Colloquia	40		
Projects	20		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Interoperability and Information Integration		
Lecturer (for lectures)		Stoimenov V. Leonid, Bogdanović D. Miloš		
Lecturer/associate (for exercises)		Bogdanović D. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Introduction to the need for information integration and the need to implement application and system interoperability. Familiarization with problems in information integration and mastery of technologies for implementing system integration and interoperability.		
Course outcomes		Theoretical and practical knowledge of concepts, methods of solution, design and implementation of basic elements of system interoperability and information integration.		
Course outline				
Theoretical teaching		Introduction to the problem of information integration. The problem of heterogeneity and ways to solve it: technical, syntactic, semantic. Architectures for information integration. Ontologies and semantic heterogeneity. Patterns for creating ontologies. Solving the problem of semantic heterogeneity, types of ontologies for information integration, mappings between ontologies. Interoperability of applications and systems. Approaches and platforms for implementing interoperability. Technologies for implementing interoperability: Enterprise Service Bus, service-oriented architectures. Standards and their importance for implementing interoperability. Open data ontologies and Web portals as access points to integrated information. Open data and semantic interoperability of systems based on open data. The concept of Big Data and architectures for storing large amounts of data, Data Warehouse, Data Lake, etc.		
Practical teaching (exercises, OFE, study and research work)		Practical implementation of simple examples of information integration that demonstrate the existence of heterogeneity problems and their resolution at the technical and syntactic levels. Using open source systems to implement solutions to heterogeneity at the syntactic and technical levels based on standards. Implementing mapping between ontologies. Implementing mechanisms for determining semantic similarity of ontologies. Implementing mechanisms for determining semantic heterogeneity of the system based on data analysis and data structure used within the system. Implementing system interoperability based on ESB and service-oriented architecture. Using open data to integrate heterogeneous systems. Determining semantic similarity of open data ontologies. Analysis of existing tools and systems and possibilities of their application.		
Textbooks/references				
1		V. E. Ferraggine, J.H. Doorn, L.C. Rivero (Eds.) Handbook of Research on Innovations in Database Technologies and Applications: Current and Future Trends (2-volumes), ISBN: 978-1-60566-242-8, Publisher: Information Science Reference, February 2009, Pages: 1,124, pp. 491-507.		
2		R. Emasri, S. Navathe, Fundamentals of Database Systems, Pearson; 7 edition (2016), ISBN-13: 9780133970777		
3		A. Sheth, Semantic Services, Interoperability and Web Applications: Emerging Concepts, 2011, ISBN: 978-1609605933, Publisher: IGI Global		
4		J. T. Pollock, R. Hodgson, Adaptive Information: Improving Business Through Semantic Interoperability, Grid Computing, and Enterprise Integration, 2004, ISBN: 978-0471488545, Publisher: John Wiley & Sons, Inc., New Jersey		
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures, auditory exercises, laboratory exercises, independent work of students on homework and projects		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		70	Oral exam	30
Colloquia				

Projects			

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Computer Animation		
Lecturer (for lectures)			Milosavljević Lj. Aleksandar, Dimitrijević M. Aleksandar		
Lecturer/associate (for exercises)			Dimitrijević M. Aleksandar		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Obligatory / Elective
Prerequisites		None			
Course objectives		Introduction to basic algorithms and techniques for computer animation.			
Course outcomes		Theoretical and practical knowledge of algorithms and techniques for computer animation. Ability to independently program graphic applications as well as to use ready-made software programs for computer animation.			
Course outline					
Theoretical teaching		Introduction to computer animation. Fundamentals of kinematics in animation. Body dynamics and physics-based animation. Skeletal animation and motion interpolation. Motion capture. Procedural animation. Facial animation and expression. Simulation of soft bodies and fabrics. Animation of fluids, smoke and fire. Crowd animation. Stylized animation and temporal-spatial optimization. Interactive animation and real-time computer games. Artificial intelligence in animation. Tools and environments in professional production. Directions and challenges in contemporary animation.			
Practical teaching (exercises, OFE, study and research work)		Introduction to computer animation software tools. User interface of Blender, an open source modeling and animation software. Fundamentals of animation in Blender (basic transformations, keyframes, smooth transitions and interpolation types, curve tracking). Armatures (single and multi-bone armatures, humanoid armatures, unconnected bones, vertex groups and field of influence, inverse kinematics). Humanoid animation based on a bone system. Corrective shaping keys.			
Textbooks/references					
1		Rick Parent et al., Computer Animation Complete, Morgan Kaufmman Publ., 2009			
2		Marcia Kuperberg et al., A Guide to Computer Animation for TV, Games, Multimedia and Web, Focal Press, 2002			
3		The Complete Guide to Blender Graphics, Blender 2.50, John M. Blain, CRC Press, 2012			
4		Blender 2.5 Character Animation Cookbook, Blender 2.50, Virgilio Vasconcelos, Packt Publishing, 2011.			
5		Introducing Character Animation with Blender 2nd ed, Blender 2.50, Tony Mullen, Sybex, 2011.			
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods		Lectures, auditory exercises, laboratory exercises, independent work of students on projects.			
Grade (maximum number of points 100)					
Pre-exam duties		Points	Final exam		Points
Activity during lectures		10	Written exam		
Exercises		40	Oral exam		40
Colloquia					
Projects		10			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Requirements Engineering		
Lecturer (for lectures)		Rančić D. Dejan, Milosavljević Lj. Aleksandar		
Lecturer/associate (for exercises)		Mihajlović T. Vladan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Introducing students to the field of requirements engineering and introducing them to the principles of requirements management as well as basic requirements engineering models.		
Course outcomes		Knowledge of basic engineering principles and models is required.		
Course outline				
Theoretical teaching		Introduction and history of the field of requirements engineering. Requirements management. Functional and non-functional requirements. Problems in requirements specification. Using standards in requirements specification. Spiral model of the requirements management process. Requirements gathering. Requirements analysis. Requirements validation. Types of requirements. Requirements characteristics. Dummy requirements. Requirements engineering methods. DFD diagrams. Relational methods. Object-oriented methods. Formal methods. Methods based on system behavior. Use-case specification. Viewpoint-based methods. Software tools for requirements engineering.		
Practical teaching (exercises, OFE, study and research work)		Requirements management systems (JIRA, Redmine, Trello, etc.) their roles and capabilities in the requirements engineering process. Good practice in organizing requirements and tracking their changes during software development. Techniques for collecting requirements and extracting requirements from text. Representing requirements using models (ULM, BMP). Tools for creating models and their characteristics. Collecting, defining and modeling requirements on real examples.		
Textbooks/references				
1		Klaus Phol, Requirements Engineering - Fundamentals, Principles, and Techniques., Springer, 2010.		
2		Klaus Phol, Chriss Rupp, Requirements Engineering Fundamentals - A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB compliant, RockyNock, 2013.		
3		E. Hull, Ken Jackson, Jeremy Dick, Requirements Engineering, Springer, 2005.		
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures, auditory exercises, independent work of students on projects.		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		10	Written exam	
Exercises			Oral exam	40
Colloquia				
Projects		50		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Advanced Computer Graphics		
Lecturer (for lectures)		Dimitrijević M. Aleksandar		
Lecturer/associate (for exercises)		Dimitrijević M. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS	6	Course status (obligatory/elective)	Obligatory / Elective	
Prerequisites	None			
Course objectives	An introduction to programming modern graphics hardware and to advanced algorithms in computer graphics.			
Course outcomes	Theoretical and practical knowledge of algorithms and techniques in modern computer graphics. Competence in independent programming and optimization of applications using modern graphics APIs.			
Course outline				
Theoretical teaching	Architecture and programming of modern graphics processors. Graphics pipeline - overview of programmable and fixed-function stages. Data transfer and communication between the CPU and GPU; types of rendering. Vertex processing. Transform feedback. Tessellation. Geometry processing. Clipping and rasterization. Fragment processing. Retrieval of rendering results. General-purpose computations on the GPU. Synchronization and interoperability. Debugging and profiling of graphics applications. Alternative geometric pipeline. Algorithms and methods for rendering optimization.			
Practical teaching (exercises, OFE, study and research work)	Comparison of application programming interfaces (APIs). Implementation of support for graphics application development in the Microsoft Visual Studio environment. Specification of attributes and primitive drawing. Implementation of vertex shaders. Implementation of transform feedback. Implementation of tessellation shaders. Implementation of geometry shaders. Management of clipping, primitive culling, and rasterization. Implementation of fragment shaders. Use of frame buffer objects (FBO). Implementation of compute shaders. Implementation of synchronization and data exchange between different APIs. APIs, tools, and environments for debugging and profiling. Implementation of task and mesh shaders. Examples of optimization techniques.			
Textbooks/references				
1	Segal, M. and Akeley, K. The OpenGL Graphics System: A Specification, Version 4.6 (Core Profile), 2022. https://registry.khronos.org/OpenGL/specs/gl/glspec46.core.pdf			
2	Kessenich, J., Baldwin, D. and Rost, R. The OpenGL Shading Language, Version 4.60.8, 2023. https://registry.khronos.org/OpenGL/specs/gl/GLSLangSpec.4.60.pdf			
3	Wolff, D. OpenGL 4 Shading Language Cookbook: Build high-quality, real-time 3D graphics with OpenGL 4.6, GLSL 4.6 and C++17, 3rd Edition, Packt Publishing, 2018. 978-1789342253			
4	Sellers, G., Wright R. Jr., Haemel, N. OpenGL Superbible: Comprehensive Tutorial and Reference 7th Edition, Addison-Wesley Professional, 2015. ISBN: 978-0672337475			
5	Kosarevsky, S., Latypov, V. 3D Graphics Rendering Cookbook: A comprehensive guide to exploring rendering algorithms in modern OpenGL and Vulkan, Packt Publishing, 2021. ISBN: 978-1838986193			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, tutorials, and independent project-based student work.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	30	Oral exam		40
Colloquia				
Projects	30			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Database Management Systems		
Lecturer (for lectures)		Stoimenov V. Leonid, Stanimirović S. Aleksandar		
Lecturer/associate (for exercises)		Stanimirović S. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Mastering knowledge of the basic concepts and principles of database management systems (DBMS) and their components. Gaining knowledge of advanced DBMS techniques (triggers, security, query optimization) and DBMS functions (replication, recovery, backups, fragmentation). Acquiring understanding of the fundamental concepts and principles of operation of advanced database systems, systems for handling large volumes of data, and large distributed systems			
Course outcomes	Theoretical and practical knowledge of database management systems, their components, and methods of use. Theoretical and practical knowledge of advanced DBMS techniques, administration, performance tuning, query optimization, recovery, backups, and data maintenance. Theoretical and practical knowledge of systems for handling large volumes of data and distributed systems supporting social networks and other large-scale systems.			
Course outline				
Theoretical teaching	Introduction to database management systems (DBMS). The role and responsibilities of the database administrator. Database management system: overview of architecture, core modules and functions, examples of such systems. Stored procedures and triggers. Query processing and optimization: concept of query optimization, relational algebra and execution plans, static and dynamic optimization, system catalog, database statistics and optimization. Indexes and index structures, multidimensional indexes, the role of indexes, and recommendations for proper use. Database system security: concept of database system security, security in DBMS, user privileges – granting and revoking (GRANT and REVOKE commands), privilege propagation, view-level security, statistical databases, DAC and MAC security mechanisms. Recovery and transactions. Replication. Backups. Distributed systems and systems for handling large volumes of data. Overview of features in open-source and commercial systems. Overview of DBMS functionalities in object-oriented databases, object-relational DBMS, and NoSQL databases.			
Practical teaching (exercises, OFE, study and research work)	The role of the database administrator, practical use of DBMS and administrative tools – practical examples and tasks. Database performance tuning, creating indexes. Query optimization – practical examples, issues, use of DBMS tools. Security in DBMS and the role of the database administrator, security at the operating system, network, hardware level, etc. – practical examples and tasks. Modern challenges for DBMS – example use of new technologies (object-relational mappers and their use: Hibernate/NHibernate, XML, etc.). NoSQL databases (concept, basic principles, classification, examples) and differences compared to traditional DBMS. Use of systems for big data processing and distributed systems.			
Textbooks/references				
	1	R.Ramakrishnan, J. Gehrke, Database Management Systems,Third Edition, 2002, ISBN-13: 978-0072465631, McGraw-Hill.		
	2	R. Elmasri, S. Navathe, Fundamentals of Database Systems, Pearson; 7 edition (2016), ISBN-13: 9780133970777		
	3	T. M. Conolly, C. E. Begg, Database Systems: A Practical Approach to Design, Implementation and Management, пето издање, 2009		
	4	G. Harrison, Next Generation Databases: NoSQLand Big Data, 1st edition, Apress, 2015, ISBN-13: 978-1484213308		
	5	Literature from the Internet on modern trends in database management systems.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, whiteboard exercises, lab exercises, independent student work on homework assignments and projects, student seminars (presentations of student work followed by discussions).			

Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises	70	Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program	Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering
Type and level of studies	Master Academic Studies
The name of the course	Ubiquitous Computing
Lecturer (for lectures)	Stojanović H. Dragan
Lecturer/associate (for exercises)	Predić B. Bratislav
Lecturer/associate (for OFE)	
Number of ECTS	6
Course status (obligatory/elective)	Elective
Prerequisites	Mobile application development, Internet of Things and services
Course objectives	Mastering theoretical and practical knowledge in the field of mobile and ubiquitous computing and the Internet of Things (IoT), as well as methods, technologies, and software tools for the design and implementation of mobile and ubiquitous systems, applications, and services.
Course outcomes	Upon successful completion of the course, the student understands the fundamental concepts, architectures, and technologies of mobile computing, ubiquitous computing, and the Internet of Things. The student is able to design and implement ubiquitous IoT systems that integrate sensors, mobile devices, communication technologies, and services. The student is capable of collecting, transmitting, processing, and analyzing data from the environment, as well as developing location-based and context-aware systems and services. The student understands and applies the principles of AIoT, edge intelligence, and cloud and edge-cloud computing, including the processing of large volumes of data. Through practical work, the student is able to develop and evaluate functional ubiquitous IoT applications while applying appropriate privacy and security mechanisms.
Course outline	
Theoretical teaching	Introduction to Mobile, Ubiquitous Computing, and the Internet of Things (IoT). Modern mobile and ubiquitous computing/communication/sensor systems and devices. Wireless and ad-hoc networks and protocols for synchronous and asynchronous communication. The Internet of Smart and Secure Things; the Web of Things. Sensor and environmental data collection, transmission, and processing. Mobile positioning technologies and location-based systems and applications. Context recognition and the development of context-aware systems and services. The Internet of Smart Things – AIoT (Artificial Intelligence of Things). IoT systems at the edge; intelligence at the edge of infrastructure. IoT systems and services in the cloud and across the edge-cloud computing continuum. Big Data management and analytics in mobile and ubiquitous systems. Privacy and security in mobile, ubiquitous, and IoT systems. Modern mobile/ubiquitous applications and IoT systems in smart cities, homes, vehicles, healthcare, mobility and transportation, industry, agriculture, and more.
Practical teaching (exercises, OFE, study and research work)	Technologies, Components, and Systems of Ubiquitous Computing and IoT. Characteristics and functions of mobile and ubiquitous IoT devices. Communication via synchronous and asynchronous protocols. Application of AI, security, and Web technologies in ubiquitous IoT systems. Methods and technologies for collecting, transmitting, and processing sensor data from the environment. Mobile positioning technologies; development of location-based services (LBS). Implementation of context recognition and application in the development of context-aware IoT systems. Implementation of smart things and the application of AI in AIoT development. Development of IoT systems at the edge infrastructure and implementation of edge intelligence. Development and deployment of IoT systems and services in the cloud and edge-cloud computing continuum. Implementation of ubiquitous IoT systems for big data processing and analytics. Methods and technologies for ensuring privacy and security in IoT systems. Development of ubiquitous applications/IoT systems in current domains: smart cities, homes, vehicles, healthcare, mobility and transportation, industry, agriculture, and more.
Textbooks/references	
1	John Krum (Ed.): Ubiquitous Computing Fundamentals. CRC Press, October 2009.
2	Valentina E. Balas, Vijender Kumar Solanki, Raghvendra Kumar, Manju Khari, Internet of Things and Big Data Analytics for Smart Generation, Springer International Publishing, 2019
3	Perry Lea, IoT and Edge Computing for Architects, 2nd Edition, Packt Publishing, 2020
4	A. King, Programming the Internet of Things: An Introduction to Building Integrated, Device-to-Cloud IoT Solutions, O'Reilly Media, 2021.
5	D. Situnayake, J. Plunkett, AI at the Edge: Solving Real-World Problems with Embedded Machine Learning, O'Reilly Media, 2023.
Number of classes of active education per week during semester/trimester/year	

Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Teaching methods:Lectures, exercises, independent student work on assignments and projects, student seminars.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	40
Colloquia		40		
Projects		20		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering Control Systems / Automatic Control		
Type and level of studies		Master Academic Studies		
The name of the course		Virtual and Augmented Reality Systems		
Lecturer (for lectures)		Milosavljević Lj. Aleksandar, Dimitrijević M. Aleksandar		
Lecturer/associate (for exercises)		Dimitrijević M. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Introducing students to the field of virtual and augmented reality and introducing them to the basic devices, algorithms, and techniques used in the implementation of these systems.		
Course outcomes		Knowledge of the basic principles, devices, techniques and algorithms used in the implementation of virtual and augmented reality systems.		
Course outline				
Theoretical teaching		Introduction to the fields of virtual and augmented reality and historical development. Architecture of virtual reality systems. Input and output devices for virtual reality. Gestural and haptic interface. Geometric modeling of virtual objects. Kinematic and physical modeling. Virtual worlds. Positioning of augmented reality in the continuum between the real world and virtual reality. Classification of augmented reality systems. Techniques for determining the observer's view in augmented reality systems. Augmented virtual environments. Areas and examples of application of virtual and augmented reality systems.		
Practical teaching (exercises, OFE, study and research work)		Practical work includes programming virtual and augmented reality elements in the Unity environment, using the Google Cardboard, Vuforia and AR Foundation libraries. Working with Google Cardboard, students develop simple virtual reality applications for mobile devices, Vuforia is used for augmented reality applications based on image recognition, while AR Foundation enables the creation of more advanced solutions in the field of augmented reality based on surface recognition and interaction with the environment in real time.		
Textbooks/references				
1		K. Stanney, Handbook of Virtual Environments: Design, Implementation, and Applications, Lawrence Erlbaum Associates, 2002.		
2		G. Burdea, P. Coiffet, Virtual Reality Technology, Wiley-IEEE, 2003.		
3		O. Bimber, R. Raskar, Spatial Augmented Reality: Merging Real and Virtual Worlds, A K Peters, 2005.		
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures, auditory exercises, laboratory exercises, independent work of students on projects.		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		30	Oral exam	40
Colloquia				
Projects		30		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering	
Type and level of studies		Master Academic Studies	
The name of the course		DevOps - Integrated Approach to Software Development	
Lecturer (for lectures)		Predić B. Bratislav	
Lecturer/associate (for exercises)		Predić B. Bratislav	
Lecturer/associate (for OFE)			
Number of ECTS	6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None		
Course objectives	The goal of this course is for students to acquire advanced knowledge and skills in the field of DevOps, including process automation, infrastructure as code management, continuous integration and software delivery, monitoring, and security. Through theoretical lectures and practical exercises, students will learn how to improve software development and operational processes, reducing delivery time while increasing the quality and security of applications.		
Course outcomes	Upon successful completion of the course, students will be able to: - Understand the core concepts and principles of the DevOps methodology. - Implement automation processes for software development, testing, and delivery. - Use tools for continuous integration and delivery (CI/CD). - Manage infrastructure using Infrastructure as Code (IaC). - Implement system monitoring and logging. - Optimize application performance and security. - Develop strategies for container orchestration and microservices architectures.		
Course outline			
Theoretical teaching	The following topics provide a balance between theoretical foundations and practical applications, enabling students to gain a comprehensive understanding of DevOps: 1. Introduction to DevOps 2. DevOps Culture and Principles 3. Continuous Integration (CI) 4. Continuous Delivery and Deployment (CD) 5. Version Control and Collaboration 6. Infrastructure as Code (IaC) 7. Containerization and Docker 8. Container Orchestration: Kubernetes 9. Monitoring and Logging 10. Security in DevOps 11. Cloud Computing and DevOps 12. Serverless Architectures and Function as a Service (FaaS) a. Lambda, Azure Functions, Google Cloud Functions b. Benefits and Challenges of Serverless 13. Testing in a DevOps Environment 14. Incident Management and System Resilience 15. DevOps in Practice: Case Study		
Practical teaching (exercises, OFE, study and research work)	The exercises cover key practical aspects of DevOps and prepare students for work in production environments: 1. Working with Git and GitHub – Basic commands, branching, merging, and pull requests. 2. Setting up a Jenkins CI/CD Pipeline – Automating builds, testing, and deployment. 3. Creating Docker Containers – Writing Dockerfiles, building, and running containers. 4. Container Orchestration with Kubernetes – Deployment, scaling, and service management. 5. Using Terraform for Infrastructure Management – Defining cloud infrastructure as code. 6. Implementing Monitoring with Prometheus and Grafana – Collecting and visualizing metrics. 7. Security in DevOps – Working with Vault, container scanning, and secrets management. 8. AWS Lambda and Serverless Functions – Creating and executing serverless functions. 9. Automated Application Testing – Setting up Selenium tests in the CI/CD process. 10. Implementing Logging with the ELK Stack – Configuring and analyzing logs in Elasticsearch.		
Textbooks/references			
1	Kim, G., Behr, K., & Spafford, G. (2013). The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win. IT Revolution Press. ISBN: 978-0988262591		

2	Kim, G., Humble, J., Debois, P., & Willis, J. (2016). The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations. IT Revolution Press. ISBN: 978-1942788003			
3	Forsgren, N., Humble, J., & Kim, G. (2018). Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations. IT Revolution Press. ISBN: 978-1942788331			
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, auditory exercises, laboratory exercises, and student projects.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	30
Exercises		10	Oral exam	40
Colloquia				
Projects		20		

Specification for the book of courses

Study program		Computer Science and Informatics		
Type and level of studies		Master Studies		
The name of the course		Master Thesis – Study and Research Work		
Lecturer (for lectures)				
Lecturer/associate (for exercises)				
Lecturer/associate (for OFE)				
Number of ECTS	6	Course status (obligatory/elective)	Obligatory	
Prerequisites				
Course objectives	Application of basic, theoretical-methodological, scientific-professional and professional-applied knowledge and methods to solve specific problems. The student studies the problem, its structure and complexity and, based on the conducted analyses, draws conclusions about possible ways of solving it. By studying the literature, the student becomes familiar with the methods intended for solving similar tasks and engineering practice in solving them.			
Course outcomes	Enabling students to independently apply previously acquired knowledge from various areas they have studied, in order to observe the structure of a given problem and its systemic analysis in order to draw conclusions about possible directions for its solution. Through the independent use of literature, students expand their knowledge by studying various methods and works related to similar issues. In this way, students develop the ability to conduct analyses and identify problems within the given issue. Through the practical application of acquired knowledge, students develop the ability to perceive the place and role of engineers in the chosen area, the need for cooperation with other professions and teamwork.			
Course outline				
Theoretical teaching	It is formed individually in accordance with the needs of a specific diploma-master's thesis, its complexity and structure. The student, according to his affinities and preferences, chooses the field of study work, or rather the subject teacher from the list of teachers in the study program who defines a specific task for him. The student studies professional literature, professional and scientific works that deal with similar topics, performs analyses in order to find a solution to a specific task or performs certain experiments in the laboratory. The study work also includes active monitoring of primary knowledge, organization and performance of experiments, numerical simulations and statistical data processing, and preparation of a seminar paper in a narrow scientific and educational field to which the topic of the independent research work belongs.			
Practical teaching (exercises, OFE, study and research work)				
Textbooks/references				
1				
2				
3				
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
			12	
Teaching methods				
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	50
Colloquia				
Projects		50		

Specification for the book of courses

Study program		Computer Science and Informatics	
Type and level of studies		Master Studies	
The name of the course		Professional Practice	
Lecturer (for lectures)			
Lecturer/associate (for exercises)			
Lecturer/associate (for OFE)			
Number of ECTS	3	Course status (obligatory/elective)	Obligatory
Prerequisites			
Course objectives	Familiarization with the work process in the company where the professional internship is conducted, its objectives, and organizational units. Getting to know the team and the project that the student joins as part of their professional internship, selected in accordance with the elective study area (module) chosen by the student. Understanding the company's workflow, business processes, work-related risks, and participating in design, documentation development, or quality control, in alignment with the work process and the possibilities of the work environment.		
Course outcomes	Enhancing the student's ability to integrate into the workforce upon completing their studies. Developing responsibility, a professional approach to work, and teamwork communication skills. Supplementing the theoretical knowledge acquired in the academic program with practical insight into the issues studied within the student's field. Utilizing the expertise of professionals employed at the institution where the internship is conducted to broaden practical knowledge and motivate students. Gaining a clear understanding of how the acquired knowledge and skills from the academic program can be applied in practice.		
Course outline			
Theoretical teaching			
Practical teaching (exercises, OFE, study and research work)	The content of the professional practice is fully aligned with its objectives. The student becomes familiar with the company's structure and business goals, adapts their involvement to the chosen field of study, and duly fulfills work responsibilities in accordance with the duties of employees in the company. The student describes their own engagement during the internship and provides a critical reflection on their experience, knowledge, and skills acquired throughout the practice.		
Textbooks/references			
	1		
	2		
	3		
	4		
	5		
Number of classes of active education per week during semester/trimester/year			
Lectures	Exercises	OFE	Study and research work
			Other classes
			6
Teaching methods	The student generally independently selects a company from the public, private, or state sector in which they will complete their professional internship. The internship can also be carried out abroad, in which case the student additionally improves their foreign language skills. Based on the student's proposal, the head of the selected field/module approves the internship at the desired institution and, upon request, issues a written referral for the internship to the person responsible for conducting the practice at the given institution. Upon completion of the internship, and based on the student's report and confirmation from the responsible individual—verified with a signature and the company seal—the student is awarded 3 ECTS credits for the completed practice.		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises	70	Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program		Computer Science and Informatics	
Type and level of studies		Master Studies	
The name of the course		Master Thesis	
Lecturer (for lectures)			
Lecturer/associate (for exercises)			
Lecturer/associate (for OFE)			
Number of ECTS	9	Course status (obligatory/elective)	Obligatory
Prerequisites			
Course objectives	The Master's thesis aims to consolidate, confirm and practically apply the knowledge acquired during the Master's academic studies. The student is given the opportunity to demonstrate the ability to independently carry out a project, which can be of a practical, research or theoretical-methodological nature. The student also gains experience in presenting his/her work through a written form and an oral presentation during the thesis defense.		
Course outcomes	Ability to lead an independent project, ability to formulate and analyze problems, critically review possible solutions, review literature in a given field. Application of acquired engineering and design knowledge and skills to solve problems, taking into account the complexity, costs, reliability and efficiency of the solution. Ability to write a paper in a given form. Ability to clearly explain the completed project through an oral defense of the paper.		
Course outline			
Theoretical teaching	A master's thesis is an independent research, practical or theoretical methodological work of a student in accordance with the level of study, in which he becomes acquainted with a narrow field through a literature review and adopts the research methodology, or design, necessary for the preparation of the thesis. Through the preparation of the thesis, the student applies practical and theoretical knowledge acquired during his studies. The written thesis usually contains an introductory chapter, a definition of the problem, a review of the field and existing solutions, a proposal and description of the solution, a conclusion and literature. A public oral defense of the thesis is organized before a committee of three members, one of whom is the thesis mentor. During the oral defense, the candidate explains the results of his work, and then answers questions from the committee members, thereby demonstrating the candidate's ability to orally present his project.		
Practical teaching (exercises, OFE, study and research work)			
Textbooks/references			
1			
2			
3			
4			
5			
Number of classes of active education per week during semester/trimester/year			
Lectures	Exercises	OFE	Study and research work
			2
Teaching methods			
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	70
Exercises		Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering	
Type and level of studies			Master Academic Studies	
The name of the course			Advanced Software Engineering	
Lecturer (for lectures)			Rančić D. Dejan, Milosavljević Lj. Aleksandar	
Lecturer/associate (for exercises)			Mihajlović T. Vladan	
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Basic knowledge of object-oriented programming, databases, and software engineering fundamentals.			
Course objectives	Mastering advanced methods for software development and evolution and methods for measuring the quality of software products and processes.			
Course outcomes	The student is equipped to apply agile methodologies in software planning and development. The student is capable of using advanced version control and project management systems. The student knows how to analyze metrics and apply techniques for continuous software quality improvement.			
Course outline				
Theoretical teaching	A brief overview and history of software engineering. Software evolution. Software metrics. Software cost estimation. Quality management. Software process improvement. Agile methods for software development. Introduction to Scrum. User stories. Agile planning and estimating. Configuration management. Code control systems. Software reuse. Component-based software engineering. Prototyping. Software metrics. Development of safety-sensitive software. Development of real-time software.			
Practical teaching (exercises, OFE, study and research work)	Introduction to the SVN and GIT code control systems and the basic operations they offer. Best practices in organizing branches in GIT code in software development. Solving code merging problems. Introduction to project management and bug tracking support systems Redmine and JIRA. Support for agile software development in JIRA. Introduction to current software development technologies through seminars.			
Textbooks/references				
	1	Ian Sommerville, Engineering Software Products: An Introduction to Modern Software Engineering, Pearson, 2020.		
	2	Roger Pressman, Bruce Maxim, Software Engineering: A Practitioner's Approach, 9th ed., McGraw-Hill, 2020.		
	3	Robert C. Martin, Clean Agile: Back to Basics, Prentice Hall, 2019.		
	4			
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, auditory exercises, independent work of students on projects.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		30	Oral exam	40
Colloquia				
Projects		30		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Intelligent Systems		
Lecturer (for lectures)		Stoimenov V. Leonid, Frtunić-Gligorijević B. Milena		
Lecturer/associate (for exercises)		Frtunić-Gligorijević B. Milena		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	To provide students with insight into advanced artificial intelligence techniques, to present current problems and possible solutions for the implementation of intelligent systems. To introduce students to the importance and process of prototype realization, i.e. MVP (Minimum Valuable Product). Theoretical and practical knowledge of various areas of AI: the importance of computer vision, communication and planning for the implementation of intelligent systems, problems of inference with unreliable knowledge, elements of natural language processing, speech processing and synthesis, machine learning and neural networks. To present students with possible applications of intelligent systems for business applications in various real-world application domains. Presentation of existing libraries with implemented AI algorithms and possibilities of their application. Presentation of existing data sources.			
Course outcomes	At the end of the course, the student will be able to understand current problems in the implementation of intelligent systems, as well as future directions of research and development in artificial intelligence. The student will be able to respond to the challenges of selecting and designing individual parts of an intelligent system. The student will be able to recognize the problems of implementing distributed intelligent systems, the problem of semantic integration of information, and to implement some solutions using open source libraries or independently in the form of MVP.			
Course outline				
Theoretical teaching	Software engineering and artificial intelligence. The concept of MVP (Minimum Valuable Product) and how to apply it to systems based on artificial intelligence. The process of developing intelligent systems. Artificial intelligence techniques and their possible areas/domains of application. Computer vision. Communication: natural language processing. Speech recognition. Natural language understanding. Speech synthesis. Planning and planning algorithms. Probabilistic inference. Machine learning and machine learning algorithms. Neural networks and biologically inspired algorithms. Distributed intelligence and distributed inference systems. Application of intelligent systems in different domains. Examples of good practice.			
Practical teaching (exercises, OFE, study and research work)	Principles of software engineering for the development of intelligent systems. Implementation of systems in the form of an MVP (Minimum Viable Product). Introduction to algorithms and methods for computer vision. Algorithms and methods for natural language processing. Algorithms for speech recognition and synthesis. Machine learning algorithms and neural networks. Decision trees and business intelligence. Ontologies and semantic representation. Implementation and possible applications of intelligent systems. Examples of open-source libraries and their capabilities for implementing intelligent systems. Examples of data sources that can be used for the development of intelligent systems.			
Textbooks/references				
	1	S. Russell, P. Norvig: Artificial Intelligence: A Modern Approach, 4th edition, Pearson, 2020. ISBN 978-0134610993.		
	2	A. Biswas, W. Talukdar, Building Agentic AI Systems: Create intelligent, autonomous AI agents that can reason, plan, and adapt, Packt Publishing, 2025, ISBN-10 : 1803238755		
	3	Handbook on Artificial Intelligence-Empowered Applied Software Engineering, Eds. Maria Virvou, George A. Tsihrintzis, Nikolaos G. Bourbakis, Lakhmi C. Jain, Springer Cham, 2023, ISBN 978-3-031-08204-7		
	4	L. Bass, Q. Lu, I. Weber, L. Zhu, Engineering AI Systems: Architecture and DevOps Essentials, 2025, Addison-Wesley Professional, ISBN: 9780138261542		
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0

Teaching methods	Lectures, auditory exercises, laboratory exercises, independent work of students on homework and projects		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises	40	Oral exam	30
Colloquia			
Projects	30		

Specification for the book of courses

Study program	Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering Control Systems / Automatic Control
Type and level of studies	Master Academic Studies
The name of the course	Geospatial Information Systems
Lecturer (for lectures)	Stojanović H. Dragan, Predić B. Bratislav
Lecturer/associate (for exercises)	Predić B. Bratislav
Lecturer/associate (for OFE)	
Number of ECTS	6 Course status (obligatory/elective) Obligatory / Elective
Prerequisites	Databases, Information systems
Course objectives	Mastering the knowledge, methods, and technologies necessary for the design and implementation of geospatial information systems (GIS) and applications, as well as the application of spatial data science.
Course outcomes	Upon successful completion of the course, the student understands the fundamental concepts, methods, and technologies of geospatial information systems and spatial data science. The student is able to collect, model, store, process, and analyze raster, vector, and spatiotemporal geospatial data. The student is capable of designing and implementing geospatial databases, applying spatial indexing structures, and developing efficient geospatial queries and analyses. The student is able to apply standards and technologies for developing Web GIS, OGC services, mobile GIS applications, and location-based services. Through practical work, the student is able to use modern GIS tools, libraries, and platforms for visualization, spatial analysis, mobility analysis, and processing of large-scale geospatial data.
Course outline	
Theoretical teaching	Introduction to Geospatial Information Systems (GIS). Geospatial coordinate systems. Methods and systems for positioning and collecting geospatial data: GPS, remote sensing, aerial and orthophoto imaging, RADAR, LIDAR, conversion of existing data, OpenStreetMap (OSM), etc. Raster representation of geospatial data, formats, and processing functions. Vector representation of geospatial data, formats, functions, and algorithms for their processing and analysis. Geospatial databases. Indexing structures and methods for accessing geospatial data. Time in GIS and management of spatiotemporal data. Visualization of geospatial data, interaction methods, and visual analytics. Analysis of geospatial data and spatial data science. Analysis of mobile data, trajectories, spatial networks – mobility data science. Processing and analysis of large geospatial data sets. Specifications and standards in geospatial data and GIS (OGC, ISO TC 211). Web GIS and OGC-specified GI services (WMS, WFS, WCS, WPS, etc.). Mobile GIS and location-based services.
Practical teaching (exercises, OFE, study and research work)	Basic Functions and Components of GIS (QGIS, ArcGIS Pro). Data representation in different coordinate systems. Methods and technologies for managing geospatial data in GIS. Application of libraries and functions for processing and analyzing raster data (GDAL, rasterio). Application of libraries and functions for processing and analyzing vector data (GEOS, shapely). Design and implementation of geospatial databases (PostGIS/PostgreSQL). Application of indexing structures in processing geospatial queries (PostGIS). Processing and analysis of spatiotemporal data and GIS. Visualization and visual analysis of geospatial data (Leafmap, Folium, etc.). Geospatial data analysis and implementation of spatial data science (SAGA, GeoPandas, PySal). Mobility and trajectory data analysis (MovingPandas, scikit-mobility). Systems for processing and analyzing large geospatial data (GeoSpark, Sedona, Mosaic). Implementation and application of OGC Web GIS services (GeoServer, pygeoapi). Development of Web GIS applications using libraries and frameworks (Leaflet, OpenLayers). Development of mobile GIS applications and location-based services on Android/iOS platforms.
Textbooks/references	
1	Kang-tsung Chang, Introduction to Geographic Information Systems, 9th Edition, McGraw-Hill Science, 2019
2	Matt Duckham, Qian (Chayn) Sun, Michael F. Worboys, GIS: A Computing Perspective, 3rd edition, CRC Press, 2024.
3	Michael Dorman, Anita Graser, Jakub Nowosad, Robin Lovelace, Geocomputation with Python, Chapman and Hall/CRC, 2025.
4	Michael Dorman, Introduction to Web Mapping, Chapman and Hall/CRC, 2020.

5 Shashi Shekhar, Sanjay Chawla, Spatial Databases: A Tour, Pearson; 1st edition, 2003				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, exercises, independent student work on assignments and projects, student seminars.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	40
Colloquia		40		
Projects		20		

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Advanced Web Technologies		
Lecturer (for lectures)			Petković M. Ivan		
Lecturer/associate (for exercises)			Petković M. Ivan		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Obligatory / Elective
Prerequisites	None				
Course objectives	Introduction to advanced concepts of web application development, including the use of high-performance and memory-safe programming languages.				
Course outcomes	Students should acquire sufficient theoretical and practical knowledge to design and implement complex, high-performance, and secure web applications using the Rust programming language				
Course outline					
Theoretical teaching	High-performance languages. Classification of languages based on memory management. The problem of memory safety. Advantages and disadvantages of object-oriented and functional programming. Core concepts of Rust. Variable ownership and borrow checking. Mutable and immutable references. Structs. Enumerations. Handling null values. Returning results. Variable lifetimes. Inheritance and composition. Smart pointers. Lambda functions. Closures and functional programming. Asynchronous programming. Metaprogramming using macros. Rust and WebAssembly. Full stack Rust web applications.				
Practical teaching (exercises, OFE, study and research work)	Development tools and environment for Rust. Package manager - Cargo. Variables and references. Functions. Data structures and behaviors (traits). Enumerations. Option and null values. Result and function return handling. Introduction to Rust API implementation environment. Controllers. Validation using attributes. Database access via SQL mappers (sqlx, etc.). SQL Mappers or ORM. Database migrations. Query validation during compilation. Writing tests. Defining multiple environments (test, production). Exception and error handling. Best practices for writing Rust web applications.				
Textbooks/references					
1	Steve Klabnik & Carol Nichols, The Rust Programming Language, No Starch Press, 2019				
2	Jim Blandy, Jason Orendorff, Leonora F. Tindall, Programming Rust: Fast, Safe Systems Development (2nd Edition), O'Reilly Media, 2021				
3	Jon Gjengset, Rust for Rustaceans: Idiomatic Programming for Experienced Developers, No Starch Press, 2021				
4	Bastian Gruber, Rust Web Development, Packt Publishing, 2023				
5	Luca Palmeri, Zero To Production In Rust: An introduction to backend development, 2022				
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods	Lectures and computer-based demonstration exercises. Independent hands-on practice sessions.				
Grade (maximum number of points 100)					
Pre-exam duties		Points	Final exam		Points
Activity during lectures		5	Written exam		
Exercises		20	Oral exam		30
Colloquia					
Projects		45			

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Advanced Operating Systems		
Lecturer (for lectures)			Stanimirović S. Aleksandar, Stojanović H. Dragan		
Lecturer/associate (for exercises)			Stanimirović S. Aleksandar		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Elective
Prerequisites		Operating systems, Distributed systems			
Course objectives		Understanding the technologies, development trends, as well as the design and implementation of modern operating systems and system software development.			
Course outcomes		Upon successful completion of the course, the student understands advanced concepts, architectures, and principles of the design and implementation of modern operating systems and system software. The student is able to analyze and apply principles of kernel development and extension, as well as embedded and real-time operating systems in various computing and IoT environments. The student is capable of designing and implementing virtualized and containerized environments and applying mechanisms for container management, scheduling, and orchestration. The student is able to develop, monitor, and scale cloud-native applications using Docker and Kubernetes platforms and appropriate monitoring tools. Through practical work, the student is able to integrate modern system software, virtualization, containerization, and Infrastructure as Code technologies in the development of reliable and scalable computing systems.			
Course outline					
Theoretical teaching		Introduction to Advanced Operating Systems and System Software. Advanced concepts, design, and implementation of modern operating systems. Architecture, design, and implementation of Windows, Unix/Linux, and macOS. Development of the Linux kernel. Extending and rebuilding the Linux kernel and implementing kernel modules. Development of embedded Linux kernels. Real-time operating systems for microcontrollers. Development of IoT systems based on embedded Linux and real-time OS. Virtualization and virtual machines (VMware). Container-based virtualization (Docker). Deployment, scheduling, and orchestration of containers (Kubernetes). Development of cloud-native applications using Docker and Kubernetes. Monitoring and scaling execution of cloud-native applications. Infrastructure as Code (IaC).			
Practical teaching (exercises, OFE, study and research work)		Development of advanced concepts in operating systems and system software. Design and implementation of the Windows operating system. Implementation of Linux kernel components. Implementation of Linux kernel extensions and kernel modules. Development of embedded Linux kernel using the Yocto Project. Implementation of real-time operating systems (Zephyr, ROS). Development of IoT systems based on embedded Linux and real-time OS (Zephyr). Virtualization and virtual machines (VMware, ESXi, VirtualBox). Container-based virtualization using the Docker platform. Implementation of methods and techniques for container deployment, scheduling, and orchestration using the Kubernetes platform. Development of cloud-native applications using Docker and Kubernetes. Monitoring and scaling of cloud-native applications (Prometheus, Grafana). Infrastructure as Code (IaC) using Terraform.			
Textbooks/references					
	1	Andrew S. Tanenbaum, Herbert Bos, Modern Operating Systems, 5th edition, Pearson Education/Prentice-Hall, 2022			
	2	Robert Love, Linux Kernel Development (3rd Edition), Addison-Wesley Professional; 2010			
	3	Frank Vasquez, Chris Simmonds, Mastering Embedded Linux Development, 4th Edition, Packt Publishing, 2025.			
	4	Ashley Davis, Bootstrapping Microservices with Docker, Kubernetes, and Terraform, Manning Publications, 2021.			
	5				
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods	Lectures, exercises, independent student work on assignments and projects, student seminars.				
Grade (maximum number of points 100)					

Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises		Oral exam	40
Colloquia	40		
Projects	20		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Interoperability and Information Integration		
Lecturer (for lectures)		Stoimenov V. Leonid, Bogdanović D. Miloš		
Lecturer/associate (for exercises)		Bogdanović D. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Introduction to the need for information integration and the need to implement application and system interoperability. Familiarization with problems in information integration and mastery of technologies for implementing system integration and interoperability.		
Course outcomes		Theoretical and practical knowledge of concepts, methods of solution, design and implementation of basic elements of system interoperability and information integration.		
Course outline				
Theoretical teaching		Introduction to the problem of information integration. The problem of heterogeneity and ways to solve it: technical, syntactic, semantic. Architectures for information integration. Ontologies and semantic heterogeneity. Patterns for creating ontologies. Solving the problem of semantic heterogeneity, types of ontologies for information integration, mappings between ontologies. Interoperability of applications and systems. Approaches and platforms for implementing interoperability. Technologies for implementing interoperability: Enterprise Service Bus, service-oriented architectures. Standards and their importance for implementing interoperability. Open data ontologies and Web portals as access points to integrated information. Open data and semantic interoperability of systems based on open data. The concept of Big Data and architectures for storing large amounts of data, Data Warehouse, Data Lake, etc.		
Practical teaching (exercises, OFE, study and research work)		Practical implementation of simple examples of information integration that demonstrate the existence of heterogeneity problems and their resolution at the technical and syntactic levels. Using open source systems to implement solutions to heterogeneity at the syntactic and technical levels based on standards. Implementing mapping between ontologies. Implementing mechanisms for determining semantic similarity of ontologies. Implementing mechanisms for determining semantic heterogeneity of the system based on data analysis and data structure used within the system. Implementing system interoperability based on ESB and service-oriented architecture. Using open data to integrate heterogeneous systems. Determining semantic similarity of open data ontologies. Analysis of existing tools and systems and possibilities of their application.		
Textbooks/references				
1		V. E. Ferraggine, J.H. Doorn, L.C. Rivero (Eds.) Handbook of Research on Innovations in Database Technologies and Applications: Current and Future Trends (2-volumes), ISBN: 978-1-60566-242-8, Publisher: Information Science Reference, February 2009, Pages: 1,124, pp. 491-507.		
2		R. Emasri, S. Navathe, Fundamentals of Database Systems, Pearson; 7 edition (2016), ISBN-13: 9780133970777		
3		A. Sheth, Semantic Services, Interoperability and Web Applications: Emerging Concepts, 2011, ISBN: 978-1609605933, Publisher: IGI Global		
4		J. T. Pollock, R. Hodgson, Adaptive Information: Improving Business Through Semantic Interoperability, Grid Computing, and Enterprise Integration, 2004, ISBN: 978-0471488545, Publisher: John Wiley & Sons, Inc., New Jersey		
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures, auditory exercises, laboratory exercises, independent work of students on homework and projects		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		70	Oral exam	30
Colloquia				

Projects			

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Computer Animation		
Lecturer (for lectures)			Milosavljević Lj. Aleksandar, Dimitrijević M. Aleksandar		
Lecturer/associate (for exercises)			Dimitrijević M. Aleksandar		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Obligatory / Elective
Prerequisites		None			
Course objectives		Introduction to basic algorithms and techniques for computer animation.			
Course outcomes		Theoretical and practical knowledge of algorithms and techniques for computer animation. Ability to independently program graphic applications as well as to use ready-made software programs for computer animation.			
Course outline					
Theoretical teaching		Introduction to computer animation. Fundamentals of kinematics in animation. Body dynamics and physics-based animation. Skeletal animation and motion interpolation. Motion capture. Procedural animation. Facial animation and expression. Simulation of soft bodies and fabrics. Animation of fluids, smoke and fire. Crowd animation. Stylized animation and temporal-spatial optimization. Interactive animation and real-time computer games. Artificial intelligence in animation. Tools and environments in professional production. Directions and challenges in contemporary animation.			
Practical teaching (exercises, OFE, study and research work)		Introduction to computer animation software tools. User interface of Blender, an open source modeling and animation software. Fundamentals of animation in Blender (basic transformations, keyframes, smooth transitions and interpolation types, curve tracking). Armatures (single and multi-bone armatures, humanoid armatures, unconnected bones, vertex groups and field of influence, inverse kinematics). Humanoid animation based on a bone system. Corrective shaping keys.			
Textbooks/references					
1		Rick Parent et al., Computer Animation Complete, Morgan Kaufmman Publ., 2009			
2		Marcia Kuperberg et al., A Guide to Computer Animation for TV, Games, Multimedia and Web, Focal Press, 2002			
3		The Complete Guide to Blender Graphics, Blender 2.50, John M. Blain, CRC Press, 2012			
4		Blender 2.5 Character Animation Cookbook, Blender 2.50, Virgilio Vasconcelos, Packt Publishing, 2011.			
5		Introducing Character Animation with Blender 2nd ed, Blender 2.50, Tony Mullen, Sybex, 2011.			
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods		Lectures, auditory exercises, laboratory exercises, independent work of students on projects.			
Grade (maximum number of points 100)					
Pre-exam duties		Points	Final exam		Points
Activity during lectures		10	Written exam		
Exercises		40	Oral exam		40
Colloquia					
Projects		10			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Requirements Engineering		
Lecturer (for lectures)		Rančić D. Dejan, Milosavljević Lj. Aleksandar		
Lecturer/associate (for exercises)		Mihajlović T. Vladan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Introducing students to the field of requirements engineering and introducing them to the principles of requirements management as well as basic requirements engineering models.		
Course outcomes		Knowledge of basic engineering principles and models is required.		
Course outline				
Theoretical teaching		Introduction and history of the field of requirements engineering. Requirements management. Functional and non-functional requirements. Problems in requirements specification. Using standards in requirements specification. Spiral model of the requirements management process. Requirements gathering. Requirements analysis. Requirements validation. Types of requirements. Requirements characteristics. Dummy requirements. Requirements engineering methods. DFD diagrams. Relational methods. Object-oriented methods. Formal methods. Methods based on system behavior. Use-case specification. Viewpoint-based methods. Software tools for requirements engineering.		
Practical teaching (exercises, OFE, study and research work)		Requirements management systems (JIRA, Redmine, Trello, etc.) their roles and capabilities in the requirements engineering process. Good practice in organizing requirements and tracking their changes during software development. Techniques for collecting requirements and extracting requirements from text. Representing requirements using models (ULM, BMP). Tools for creating models and their characteristics. Collecting, defining and modeling requirements on real examples.		
Textbooks/references				
1		Klaus Phol, Requirements Engineering - Fundamentals, Principles, and Techniques., Springer, 2010.		
2		Klaus Phol, Chriss Rupp, Requirements Engineering Fundamentals - A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB compliant, RockyNock, 2013.		
3		E. Hull, Ken Jackson, Jeremy Dick, Requirements Engineering, Springer, 2005.		
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures, auditory exercises, independent work of students on projects.		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		10	Written exam	
Exercises			Oral exam	40
Colloquia				
Projects		50		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Advanced Computer Graphics		
Lecturer (for lectures)		Dimitrijević M. Aleksandar		
Lecturer/associate (for exercises)		Dimitrijević M. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS	6	Course status (obligatory/elective)	Obligatory / Elective	
Prerequisites	None			
Course objectives	An introduction to programming modern graphics hardware and to advanced algorithms in computer graphics.			
Course outcomes	Theoretical and practical knowledge of algorithms and techniques in modern computer graphics. Competence in independent programming and optimization of applications using modern graphics APIs.			
Course outline				
Theoretical teaching	Architecture and programming of modern graphics processors. Graphics pipeline - overview of programmable and fixed-function stages. Data transfer and communication between the CPU and GPU; types of rendering. Vertex processing. Transform feedback. Tessellation. Geometry processing. Clipping and rasterization. Fragment processing. Retrieval of rendering results. General-purpose computations on the GPU. Synchronization and interoperability. Debugging and profiling of graphics applications. Alternative geometric pipeline. Algorithms and methods for rendering optimization.			
Practical teaching (exercises, OFE, study and research work)	Comparison of application programming interfaces (APIs). Implementation of support for graphics application development in the Microsoft Visual Studio environment. Specification of attributes and primitive drawing. Implementation of vertex shaders. Implementation of transform feedback. Implementation of tessellation shaders. Implementation of geometry shaders. Management of clipping, primitive culling, and rasterization. Implementation of fragment shaders. Use of frame buffer objects (FBO). Implementation of compute shaders. Implementation of synchronization and data exchange between different APIs. APIs, tools, and environments for debugging and profiling. Implementation of task and mesh shaders. Examples of optimization techniques.			
Textbooks/references				
1	Segal, M. and Akeley, K. The OpenGL Graphics System: A Specification, Version 4.6 (Core Profile), 2022. https://registry.khronos.org/OpenGL/specs/gl/glspec46.core.pdf			
2	Kessenich, J., Baldwin, D. and Rost, R. The OpenGL Shading Language, Version 4.60.8, 2023. https://registry.khronos.org/OpenGL/specs/gl/GLSLangSpec.4.60.pdf			
3	Wolff, D. OpenGL 4 Shading Language Cookbook: Build high-quality, real-time 3D graphics with OpenGL 4.6, GLSL 4.6 and C++17, 3rd Edition, Packt Publishing, 2018. 978-1789342253			
4	Sellers, G., Wright R. Jr., Haemel, N. OpenGL Superbible: Comprehensive Tutorial and Reference 7th Edition, Addison-Wesley Professional, 2015. ISBN: 978-0672337475			
5	Kosarevsky, S., Latypov, V. 3D Graphics Rendering Cookbook: A comprehensive guide to exploring rendering algorithms in modern OpenGL and Vulkan, Packt Publishing, 2021. ISBN: 978-1838986193			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, tutorials, and independent project-based student work.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	30	Oral exam		40
Colloquia				
Projects	30			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Computer Network Security		
Lecturer (for lectures)		Milovanović I. Emina, Čirić M. Vladimir		
Lecturer/associate (for exercises)		Milovanović I. Emina, Čirić M. Vladimir		
Lecturer/associate (for OFE)				
Number of ECTS	6	Course status (obligatory/elective)	Obligatory / Elective	
Prerequisites	None			
Course objectives	The aim of the course is for the students to gain insight into the potential security weaknesses and obtain the basic knowledge to enhance the security, as well as to gain the knowledge and skills to apply security techniques in order to increase the security of computer networks			
Course outcomes	It is expected for the students to be able to analyze security aspects of computer networks, and to be able to plan the strategy to increase security. It is also expected that the student has a knowledge to use tools, as well as to implement security protocols in order to increase security of the network			
Course outline				
Theoretical teaching	Security objectives. Security mechanisms and services. Information theory and security. Perfectly secure ciphers. Cryptanalytic attacks. Classical cryptosystems. Historical overview of ciphers. Block ciphers. Structure of the Fiestel cipher. DES. Weaknesses of DES. 3DES. Basic concepts of number theory and finite fields. Galois fields of type GF(2 ⁿ). Modular polynomial arithmetic. AES. Cipher operation modes: ECB, CBC, CFB, OFB, CTR. Public key cryptography. RSA. Elliptic curve cryptography (ECC). Data integrity protection. Hash functions. Authentication techniques: HMAC, CBC-MAC, NMAC. Key generation and exchange protocols. Security protocols for data protection, integrity and authentication (Kerberos, SSL/TLS. IPsec, VPN, PK, PGP, S/MIME). Application of firewall tools in system protection. Attacks. Computer viruses. Security of wireless computer networks			
Practical teaching (exercises, OFE, study and research work)	Work on examples through a set of laboratory exercises. Network equipment protection. Implementation of cryptographic algorithms. Implementation of security protocols at the network, transport and application levels. Implementation of SSL/TLS and SSH protocols. RADIUS and AAA servers. Firewall. Configuration of virtual private networks (VPN). Protection of wireless networks. Typical attacks. Implementation of security protocols in OpenSSL. Secure email exchange.			
Textbooks/references				
1	William Stallings, "Cryptography and Network Security: Principles and Practice (7th Edition)", Prentice hall, 2017			
2	William Stallings – Network Security Essentials: Applications and Standards (6th Edition, 2023)			
3	W. Richard Stevens, Kevin R. Fall – TCP/IP Illustrated, Volume 1: The Protocols (2nd Edition, 2011)			
4	Michael Watkins, Kevin Wallace, "CCNA Security", Cisco Press, 2008, ISBN 9781587202209			
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, exercises, programming assignments			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		30
Exercises		Oral exam		30
Colloquia				
Projects	40			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Secure Software Design		
Lecturer (for lectures)		Janković S. Dragan, Rajković J. Petar		
Lecturer/associate (for exercises)		Rajković J. Petar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The main course objective is introducing students with basic concepts and applied techniques used for secure code development			
Course outcomes	The students have to be able to identify the main problems related to software solutions security and to apply learned resolution techniques.			
Course outline				
Theoretical teaching	Introduction. Passwords. Pseudorandom numbers. Pseudorandom number generators. Buffer Overflow problem and ways to overcome it. Safety increase techniques. Shared resources deadlock problems. Input validation. SQL injection. Cryptography elements. Authentication protocols. Software configurability. Processing sensitive data. Memory management (allocation, deallocation). Viruses and malicious software.			
Practical teaching (exercises, OFE, study and research work)	Weak points in shell scripts. Password coding. Pseudorandom numbers based applications. Implementation of random generators. Methods for the Buffer Overflow problem overcoming. Access control matrix implementation. Thread management. Methods against file level deadlocks. Input validation. Authentication protocols . Using cryptographic algorithms in software. Configurations - design and using. Problems with memory allocation and de-allocation.			
Textbooks/references				
1	Loren Kohnfeler, Designing Secure Software: A Guide for Developers, No Starch Press, 2021.			
2	Peter Gutmann: Cryptographic Security Architecture, Design and Implementation, ISBN 038795387-6, Springer, 2003.			
3	Michael Howard, David LeBlanc: Writing Secure Code, ISBN 073561722-8, Microsoft Press, 2002.			
4	Erik Fretheim, Marie Deschene, Secure Software Systems, Jones&Bartlett Learning, 2023.			
5	John Viega, Gary McGraw: Building Secure Software, ISBN 020172152-X, Addison-Wesley, 2001.			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, Auditive exercises, Laboratory exercises			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	30
Exercises			Oral exam	20
Colloquia				
Projects		50		

Specification for the book of courses				
Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Digital Forensics		
Lecturer (for lectures)		Rančić D. Dejan, Predić B. Bratislav		
Lecturer/associate (for exercises)		Predić B. Bratislav		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Introduction to the process of identifying, preserving, and analyzing digital evidence, as well as preparing it for presentation in court in a forensically sound manner.		
Course outcomes		Theoretical and practical knowledge of the process of identifying, preserving, and analyzing digital evidence. Familiarity with software and hardware tools used in digital forensics. Understanding of the legal aspects of digital forensics. Knowledge of the fundamental principles, rules, and methodologies of digital forensics.		
Course outline				
Theoretical teaching		Introduction to Digital Forensics – Fundamentals and importance in the modern ICT environment. Legal Framework and Ethics – Legality of collecting and using digital evidence. Forensic Methodology – Standards and procedures for examining digital systems. Evidence Identification and Preservation – Methods for protecting data integrity. Digital Evidence Analysis – Techniques for interpreting and correlating data. Reporting and Presentation – Forensically sound preparation of evidence. Data Recovery – Restoring deleted or damaged files. Decryption – Techniques for breaking encryption and accessing data. OS and File System Forensics – Analyzing traces in system structures. Network Forensics – Monitoring and analyzing network traffic. Mobile Forensics – Examination of smartphones and tablets. Tools and Environments – Hands-on work with EnCase, FTK, VMware, and other tools.		
Practical teaching (exercises, OFE, study and research work)		Protection and Cloning of Digital Evidence – Practical application of tools for preserving the original state of media. Forensic File System Analysis – Examination of NTFS and EXT structures to uncover traces of activity. Deleted Data Recovery – Use of specialized software to restore deleted files. System Log and Timeline Analysis – Reconstruction of events based on log files. Network Traffic Examination – Use of tools like Wireshark to monitor and analyze network communication. Mobile Device Forensics – Extraction and analysis of data from Android and iOS devices. Password Cracking and Decryption – Application of brute-force and dictionary attacks on protected data. Creation and Verification of Forensic Report – Writing reports based on analyzed evidence. Use of EnCase and FTK Tools – Basic operations within leading forensic software platforms. Court Testimony Simulation – Presentation of evidence and expert witnessing in a simulated legal proceeding.		
Textbooks/references				
	1	Digital Forensics, Investigation, and Response (4th Edition), Chuck Easttom, ISBN: 978-1284226065		
	2	Advances in Digital Forensics XX, Elizabeth Kurkowski and Sujeet Shenoj, ISBN 978-3031710247		
	3	Cybercrime and Digital Forensics: An Introduction (3rd Edition), Thomas J. Holt, Adam M. Bossler, Kathryn C. Seigfried-Spellar, ISBN 978-0367360078		
	4			
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures, auditory exercises, and student projects		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		10	Written exam	
Exercises		50	Oral exam	40
Colloquia				
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Cloud Computing		
Lecturer (for lectures)		Stojanović M. Natalija, Ćirić M. Vladimir		
Lecturer/associate (for exercises)		Stojanović M. Natalija		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The aim of the course is for students to gain insight into virtualization concepts and cloud system architecture, as well as service-oriented principles of system design and application development in cloud computing environments.			
Course outcomes	Upon successful completion of the course, the student will be able to explain the fundamental principles of virtualization, cloud computing, and high-performance computing. The student will be able to analyze requirements and design basic cloud infrastructure, apply virtualization, containerization, and orchestration technologies, and use IaaS, PaaS, and FaaS services. The student will be able to apply MapReduce, Hadoop, YARN, and Spark for distributed processing and analysis of big data, as well as technologies for distributed data storage. The student will be able to deploy, configure, and monitor the execution of distributed applications on clusters and in the cloud, and analyze and improve their performance and scalability.			
Course outline				
Theoretical teaching	Hardware virtualization. Cloud layers and types. Components and design of cloud infrastructure. Cloud infrastructure management. IaaS and PaaS providers. Migration. Amazon Web Services (AWS). Application containerization. Docker. Orchestration. Kubernetes. High-performance computing on computer clusters and in the cloud – MapReduce concept and programming model. Methods and technologies for storing big data on clusters and in the cloud (GFS, HDFS, etc.). Systems for big data processing in the cloud – Hadoop platform. Application execution in the cloud using the YARN resource management system. Systems for processing and analyzing large volumes and streams of data – Spark platform. Deployment, configuration, execution monitoring, and performance optimization of distributed software systems on clusters and in the cloud. Function as a Service (FaaS) in the cloud (AWS Lambda, Azure Functions, OpenWhisk, OpenFaaS, etc.).			
Practical teaching (exercises, OFE, study and research work)	Virtualization and use of virtual hardware. VirtualBox, VMware. Management of virtual machines in a private cloud. VMware ESXi, vSphere. Management of virtual machines in a public cloud. Amazon AWS. Application containerization in Docker. Docker files. Container management. Kubernetes. Implementation of distributed software systems using the MapReduce programming model. Storage of big data on computer clusters and in the cloud using HDFS. Development of Hadoop MapReduce applications; MapReduce data types and formats (input and output). Implementation and management of Hadoop application execution on YARN clusters. Implementation and management of Spark application execution on computer clusters. Performance improvement and evaluation of highly distributed software systems on clusters and in the cloud. Implementation of highly distributed software systems using Functions as a Service (OpenFaaS, Azure Functions).			
Textbooks/references				
	1	Anthony T. Velte, Toby J. Velte, Robert Elsenpeter, "Cloud Computing: A Practical Approach", Mc. Graw Hill, 2010, ISBN: 978-0-07-162695-8		
	2	D. C. Marinescu, Cloud Computing: Theory and Practice, 3rd Edition, Morgan Kaufmann, 2022.		
	3	Brendan Burns, Joe Beda, Kelsey Hightower, Lachlan Evenson, "Kubernetes: Up and Running: Dive into the Future of Infrastructure", O'Reilly Media; 3rd edition, 2022.		
	4	Tom White, Hadoop: The Definitive Guide, 4th Edition, O'Reilly Media, 2015.		
	5	Y. Demchenko, J. J. Cuadrado-Gallego, O. Chertov, M. Aleksandrova, Big Data Infrastructure Technologies for Data Analytics: Scaling Data Science Applications for Continuous Growth, Springer, Cham, 2024.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0

Teaching methods	Lectures, exercises, independent student work on seminar papers and projects		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	40
Exercises	40	Oral exam	
Colloquia			
Projects	20		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Virtualization		
Lecturer (for lectures)		Tošić B. Milorad		
Lecturer/associate (for exercises)		Tošić B. Milorad		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Operating systems			
Course objectives	Introducing students to the fundamentals of virtualization, types of virtualization, and technologies for implementing virtualization.			
Course outcomes	Students are able to design, configure, and use virtualized computing environments, including virtual machines, networks, data storage, and containers. They are trained to apply Docker and Kubernetes technologies, manage virtualized resources, and develop and deploy microservice applications in virtualized and cloud infrastructure.			
Course outline				
Theoretical teaching	What is virtualization: history, conceptualization, and basic principles. Types and technologies of virtualization. Operating system-level virtualization. Containers. Server virtualization. Storage virtualization. Network virtualization. Application virtualization. Virtual network functions. Virtual machines and hypervisors. Resource management systems in virtualized environments. Architectures of virtualized systems. Virtualization in IoT and devices with limited processing power.			
Practical teaching (exercises, OFE, study and research work)	Comparative analysis of existing open-source virtualization software. Installation of virtualization software. Creating a virtual machine. Creating a virtual network. Connecting two virtual machines in a LAN. Routing the internal virtual network and providing internet access. Using Docker for server virtualization. Kubernetes. Infrastructure as a service. Infrastructure for continuous development and deployment of software solutions. Implementation of an illustrative microservices-based application on a virtualized infrastructure. Commonly used architectural patterns. Network applications in next-generation architectures (6G+).			
Textbooks/references				
	1	Khedher, Omar. Mastering OpenStack: Implement the Latest Techniques for Designing and Deploying an Operational, Production-Ready Private Cloud. 3rd ed. Packt Publishing, 2024.		
	2	Zhang, Ying. Network Function Virtualization: Concepts and Applicability in 5G Networks. John Wiley & Sons, 2017.		
	3	Kavis, Michael J. Accelerating Cloud Adoption: Optimizing the Enterprise Speed and Agility. O'Reilly Media, 2020.		
	4	Bartlett, Jonathan. Cloud Native Applications with Docker and Kubernetes: Design and Build Cloud Architecture and Applications with Microservices, EMQ, and Multi-Site Configurations. Apress, 2023.		
	5	Portnoy, Matthew. Virtualization Essentials. 3rd ed. John Wiley & Sons, 2023.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, exercises, programming assignments			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures	30	Written exam		
Exercises	30	Oral exam		40
Colloquia				
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		System Administration		
Lecturer (for lectures)		Tošić B. Milorad		
Lecturer/associate (for exercises)		Tošić B. Milorad		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Operating systems, Computer networks			
Course objectives	Introducing students to basic procedures in the administration of systems based on Linux operating system principles.			
Course outcomes	Upon completion of the course, the student should be qualified to administer systems running the Linux operating system at a level sufficient to support the operational needs of medium-sized development teams.			
Course outline				
Theoretical teaching	Linux operating system. Basic principles of the Linux operating system from a system administration perspective. Overview of available open-source Linux distributions. Command-line interface and scripting languages. User management. File system operations. Operating system exposure. Server management – configuration files. Activity monitoring – logs. Principles of technical support: basic technical support principles. Remote support. Administration of virtualized systems: main types of virtualization and available technologies. Container-based systems. Virtual machine-based systems. Management of virtualized networks.			
Practical teaching (exercises, OFE, study and research work)	The course is conducted through three smaller practical projects completed individually by each student. Student presentations on the results of their individual projects. A report on the usage of artificial intelligence services and their usefulness for problems in the field of system administration.			
Textbooks/references				
	1	Nemeth, Evi, Garth Snyder, Trent R. Hein, Ben Whaley, and Dan Mackin. UNIX and Linux System Administration Handbook. 5th ed. Addison-Wesley Professional, 2022.		
	2	Ondara, Neville Asiago. Beginning Linux SysAdmin: Getting Started with Linux System Administration. Apress, 2026.		
	3	Both, David. Using and Administering Linux: Volume 2: Zero to SysAdmin: Advanced Topics. 2nd ed. Apress, 2023.		
	4			
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, exercises, programming assignments			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures	30	Written exam		
Exercises	30	Oral exam		40
Colloquia				
Projects				

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Cryptography		
Lecturer (for lectures)			Rančić D. Dejan, Vučković V. Vladan		
Lecturer/associate (for exercises)			Dimitrijević M. Aleksandar		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Obligatory / Elective
Prerequisites	None				
Course objectives	Introducing students to the field of cryptography and familiarizing them with the basic principles, algorithms, and standards used in this field.				
Course outcomes	Theoretical and practical knowledge of the basic principles, algorithms and standards used in the field of cryptography.				
Course outline					
Theoretical teaching	Introduction to cryptography and historical overview of the field. Mathematical foundations. Encryption using symmetric keys. Stream ciphers. Block ciphers. DES – The Data Encryption Standard. AES – The Advanced Encryption Standard. Encryption using a public-private key pair. Hash functions and data integrity. Digital signature schemes. RSA signing. The hash-and-sign paradigm. Certificates and public key infrastructures. Secure Socket Layer (SSL) and Transport Layer Security (TLS) standards.				
Practical teaching (exercises, OFE, study and research work)	Practical work on programming cryptographic elements using the OpenSSL library.				
Textbooks/references					
1	J. Katz and Y. Lindell, Introduction to Modern Cryptography, 3rd ed. New York, NY, USA, 2020, Chapman and Hall/CRC				
2	A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.				
3	Mark Stamp, “Information Security Principles and Practice“, John Wiley and Sons, Inc, New Jersey, U.S.A., 2006.				
4					
5					
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods	Lectures, auditory exercises, independent work of students on projects.				
Grade (maximum number of points 100)					
Pre-exam duties		Points	Final exam		Points
Activity during lectures			Written exam		30
Exercises			Oral exam		30
Colloquia		20			
Projects		20			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		High-performance Computing		
Lecturer (for lectures)		Stojanović M. Natalija, Milentijević Z. Ivan		
Lecturer/associate (for exercises)		Stojanović M. Natalija		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		Parallel and distributed systems		
Course objectives		Understanding the concepts and technologies of high-performance computing, as well as acquiring both theoretical and practical knowledge necessary for the development and analysis of high-performance applications on modern computing architectures.		
Course outcomes		Upon successful completion of the course, students will be able to explain the fundamental concepts, methods, and techniques of High-Performance Computing (HPC) and analyze different HPC architectures and environments. Students will be able to apply programming models and technologies for parallel programming on multicore, GPU, multi-GPU, and hybrid architectures, as well as in distributed, cloud, and Edge computing environments. Students will be able to develop and configure HPC applications using appropriate technologies, such as OpenMP, CUDA, ROCm, OpenACC, MPI, Hadoop, and Spark. Students will be able to apply HPC technologies in artificial intelligence applications and other application domains. Students will be able to measure, analyze, and evaluate the performance of HPC applications, including execution time and energy consumption, and apply appropriate techniques to improve their performance, scalability, and efficiency. Students will be able to explain the fundamental principles of quantum computing and its potential applications.		
Course outline				
Theoretical teaching		Overview of advanced concepts, methods, and techniques in high performance computing. High performance computing on multicore computers with shared memory (OpenMP, OpenCL, Intel TBB). Performance optimization on many-core computing architectures such as Graphics Processing Units (GPUs) (CUDA, OpenACC, Thrust, ROCm, etc.).High performance computing in distributed environments: clusters, grids, and cloud computing (MPI, Hadoop, Spark, etc.). Achieving high performance on hybrid computing architectures. High performance computing on multi-GPU architectures. Achieving high performance on edge computing systems. Applications of high performance computing in artificial intelligence (CUDA-X, cuDNN library, etc.). Quantum computing – principles, technologies, and applications. Analysis, evaluation, and optimization of HPC system performance (execution time, energy consumption), as well as programmer effort, using appropriate technologies. Applications of HPC in image and video processing, geospatial data analysis, environmental protection, business systems, bioinformatics, etc.		
Practical teaching (exercises, OFE, study and research work)		Methods and technologies for configuring and monitoring application execution, as well as evaluating and analyzing performance. Implementation of high-performance applications on multicore computers with shared memory using OpenMP and OpenCL technologies. Implementation of high-performance applications on GPU architectures using CUDA and ROCm technologies. Implementation of high-performance applications on GPU architectures using OpenACC and Thrust technologies. Performance analysis of applications on computer clusters using MPI technology. Performance analysis of applications on clusters using Hadoop and Apache Spark technologies. Performance optimization of high-performance software systems on hybrid architectures using MPI + OpenMP.Achieving high application performance on multi-GPU architectures.Achieving high application performance applicationon edge computing systems, such as the NVIDIA Jetson Nano.Examples of applying high performance computing in artificial intelligence applications using the cuDNN library.Examples of applying HPC in AI applications using the Spark MLlib library. Implementation of an application in a selected domain, followed by performance optimization, analysis, and evaluation.		
Textbooks/references				
1		W.M. Hwu, D. Kirk and I. E. Hajj, Programming Massively Parallel Processors: A Hands-on Approach, Elsevier, fourth edition, 2023.		
2		G. Barlas, Multicore and GPU Programming An Integrated Approach, Second Edition, 2023		
3		P Czarnul, Parallel Programming for Modern High Performance Computing Systems, 2018.		

4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, exercises, independent student work on seminar papers and projects			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		50	Oral exam	30
Colloquia				
Projects		20		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Computer Simulation		
Lecturer (for lectures)		Vučković V. Vladan		
Lecturer/associate (for exercises)		Vučković V. Vladan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Mastering the basic knowledge necessary for the use of basic processes and procedures in computer modeling and simulation.			
Course outcomes	Theoretical knowledge: Mastering mathematical techniques for computer modeling and simulation; 3D modeling and programming of computer simulations.			
Course outline				
Theoretical teaching	Fundamentals of computer simulation. General approach to simulation systems. Mathematical foundations of simulation systems. Mathematical modeling of physical processes and sensor data. Stochastic and deterministic approaches in simulation. Continuous and discrete simulations. Efficient data structures and simulation algorithms. Approach via differential algebraic equations. Local and distributed simulations. Discrete event simulation (DES). Aggregate Level Simulation Protocol (ALSP), Distributed Interactive Simulation (DIS), the High Level Architecture (simulation) (HLA). Introduction to 3D software. 3D simulations in real time. Parallel algorithms in simulation systems. Hardware optimization for simulation machines.			
Practical teaching (exercises, OFE, study and research work)	Hierarchies in 3D software and expression animation. Practical 3D modeling on the computer. Use of modern 3D modeling software. Basic simulation procedures. Keyframe animation – rotations and sizes (squash and stretch). NURBS modeling - lines. NURBS modeling - shapes. Introduction to polygonal modeling. Quad modeling. Polygonal modeling of a simple character. Polygonal character modeling by task. Torso. Polygonal character modeling by task. Head: eyes, lips, ear. UV mapping. Digital sculpting. Basics of 3D materialization. Rigging – Introduction to skeletal systems. The concept of Set Driven Key. Creating and selecting attributes. Programming various simple simulator classes.			
Textbooks/references				
1	Modeling and Simulation in Engineering, Edited by Catalin Alexandru, ISBN 978-953-51-0012-6, Publisher: InTech, Published: March 07, 2012 under CC BY 3.0 license.			
2	B. P. Zeigler, A. Muzy, and E. Kofman, Theory of Modeling and Simulation: Discrete Event & Iterative System Computational Foundations, 3rd ed. London: Academic Press, 2018			
3				
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, blackboard exercises, laboratory exercises, independent student work on homework and projects, consultations.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	30	Oral exam		40
Colloquia	10			
Projects	20			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Application of Multiple Valued Logic in Representation and Processing of Digital Signals		
Lecturer (for lectures)		Radmanović M. Miloš, Rajković J. Petar		
Lecturer/associate (for exercises)		Radmanović M. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Students will acquire theoretical and practical knowledge in filed of multiple-valued logic and its application in digital system design and signal processing.			
Course outcomes	Students will learn fundamentals of multiple-valued logic, related algebraic structures, and multiple-valued functions, their representation and implementation on the different technological platforms. They will learn fundamental methods for calculation of multiple-valued functions for use in digital system design and signal processing.			
Course outline				
Theoretical teaching	1. Multiple-valued functions, 2. Applications of multiple-valued functions, 3. Functional expansions for multiple-valued functions, 4. Spectral representations for multiple-valued functions, 5. Decision diagrams for multiple-valued functions, 6. Optimization of decision diagrams, 7. GF decision diagrams, 8. KGF decision diagrams, 9. RMF decision diagrams, 10. WC decision diagrams, 11. Fast algorithms for computing spectral transforms of multiple-valued functions, 12. Algorithms for computing transforms on mCPU, 13. Algorithms for computing transforms on GPU, 14. Algorithms for computing transforms on hybrid platforms.			
Practical teaching (exercises, OFE, study and research work)	Introduction and application of specialized software for working with multiple-valued functions, program implementation for working with multiple-valued functions on graphics processors. Demonstration exercises: 1. Multiple-valued functions, 2. Functional expansions for multiple-valued functions, 3. Spectral representations for multiple-valued functions, 4. Decision diagrams for multiple-valued functions, 5. Optimization of decision diagrams, 6. GF decision diagrams, 7. KGF decision diagrams, 8. RMF decision diagrams, 9. WC decision diagrams, 10. Fast algorithms for calculating spectral transformations of multiple-valued functions, 11. Algorithms for calculating transformations on mCPU, 12. Algorithms for calculating transformations on GPU, 13. Algorithms for calculating transformations on hybrid platforms.			
Textbooks/references				
1	R.S. Stankovic, J.Astola, C. Moraga, Representations of Multiple- Valued Logic Functions , Morgan & Claypool Publishers, 2012.			
2	D. Michael Miller, Mitchell Thornton, Multiple Valued Logic: Concepts and Representations, Morgan & Claypool Publishers, 2008.			
3	G. Karniadakis, R.M. Kirby, Parallel Scientific Computing in C++ and MPI: A Seamless Approach to Parallel Algorithms and Their Implementation, Cambridge University Press, 2003.			
4	S. Cook, CUDA Programming: A Developer's Guide to Parallel Computing with GPUs, Morgan Kaufmann, 2012.			
5	Teaching material on the website: http://cs.elfak.ni.ac.rs/nastava/			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures and demonstrations using slides. Seminars and projects.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	40
Colloquia				
Projects		60		

Specification for the book of courses				
Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Semantic Web		
Lecturer (for lectures)		Tošić B. Milorad, Nejković M. Valentina		
Lecturer/associate (for exercises)		Nejković M. Valentina		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		Information Systems, Web Programming, Databases, and Artificial Intelligence.		
Course objectives		Acquisition of basic theoretical knowledge and potential application areas of the Semantic Web. Mastering fundamental programming techniques for developing Semantic Web applications at the current stage of technology development. Fostering a creative attitude among students toward possible directions for further technological advancement in this field.		
Course outcomes		A systematic approach to the application of the Semantic Web in current areas of use has been developed and adopted. Theoretical knowledge of semantics in information technologies has been acquired. Students understand what ontologies are and are capable of working with them practically, as well as applying them in the currently most attractive web applications.		
Course outline				
Theoretical teaching		Introduction: Structure, syntax, and semantics. The need for semantics on the Web. Meta-programming: Metadata, XML Schema; XSLT. RDF. RDFS. Semantics: Semantics and knowledge; Ontologies; Logics; Reasoning; Domain modeling; Context. Distributed knowledge: Classification; Knowledge-based protocols. Technologies: Tools for working with ontologies; Programming packages (APIs) for ontology management; OWL. Methodologies: Ontology engineering methodologies; Methodologies for developing semantic systems. SPARQL query language and protocol. SHACL language for constraints in SPARQL. Automated reasoning. Reasoning using SPARQL query language. Semantic systems: Semantic Web Services, Semantic Web Portals, Semantic Multi-Agent Systems. Development of applications based on knowledge graphs: Searching and analyzing large datasets, Intelligence in business applications, Intelligence in integration and development of Web applications.		
Practical teaching (exercises, OFE, study and research work)		Introduction to structure, syntax, and semantics. Exercises on XML syntax. Structuring data on the Web. Practical examples of using semantics for improved search. Differences between structured and unstructured data with examples. Meta-programming and metadata. Defining and using metadata in XML. Examples with XML schemas. XSLT transformations. Using XSLT for data presentation. Converting XML data to HTML format. RDF (Resource Description Framework). Creating RDF graphs. Practical use of RDF for resource description. Ontologies and logics. Structuring ontologies using RDF and RDFS. Introduction to logical reasoning systems. Reasoning and domain modeling. Using logical rules for reasoning. Domain modeling with ontologies. Context and distributed knowledge. Modeling context in semantic applications. Classification of distributed knowledge. Tools and APIs for working with ontologies. Introduction to the Protégé tool. Using programming packages for ontology manipulation. OWL (Web Ontology Language). Creating ontologies in OWL. Linking OWL ontologies with RDF data. Semantic systems. Practical applications of the Semantic Web in multi-agent systems. Development of semantic web services and portals. Applications and data analysis. Searching and analyzing large datasets. Examples of intelligence in business applications and web integration.		
Textbooks/references				
	1	Arp, Robert, Barry Smith, and Andrew D. Spear. Building ontologies with basic formal ontology. Mit Press, 2015.		
	2	Dietz, Jan L. G., and Hans B. F. Mulder. Enterprise Ontology: A Human-Centric Approach to Understanding the Essence of Organisation. 2nd ed. Springer, 2024.□		
	3	Keet, C. Maria. "An introduction to ontology engineering." (2018).		
	4	DuCharme, Bob. Learning SPARQL: querying and updating with SPARQL 1.1. " O'Reilly Media, Inc.", 2013.		
	5	Allemang, Dean, James Hendler, and Fabien Gandon. Semantic Web for the Working Ontologist: Effective Modeling for Linked Data, RDFS, and OWL. 3rd ed. ACM Books, 2020.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes

2	2	0	0	0
Teaching methods	Lectures; Auditory exercises; Laboratory exercises; Computer exercises; Consultations; Independent student research; Oral presentations by students on chosen/assigned topics; Active student participation in classes through the use of the interactive course website.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam	Points	
Activity during lectures	30	Written exam		
Exercises	30	Oral exam	40	
Colloquia				
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Methods and Techniques of the Image Processing		
Lecturer (for lectures)		Vučković V. Vladan		
Lecturer/associate (for exercises)		Radmanović M. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Acquiring knowledge about image processing techniques and training students to independently apply the learned techniques to solve real-world problems in the field of image processing.			
Course outcomes	Upon completion of this course, students should become familiar with the methods and techniques used in the field of image processing and acquire sufficient knowledge to independently implement applications based on image processing.			
Course outline				
Theoretical teaching	Principles of image digitization. Image quantization in the original and transformation domains. Statistical image and noise models. Image transformations. Discrete representations of image transformations. Methods for filtering in the original and transformation domains. Quantification. Image restoration: Linear filters. Image restoration and segmentation: nonlinear filters. Methods for improving image quality. Image compression. Efficient image processing algorithms. Processing scanned documents. OCR systems.			
Practical teaching (exercises, OFE, study and research work)	Study research work: preparation of seminar papers in the field of advanced image processing techniques and oral presentation and defense of the paper.			
Textbooks/references				
1	L. Yaroslavsky, Digital Holography and Digital Image Processing: Principles, Methods, Algorithms, Kluwer Academic, 2004.			
2	Yaroslavsky, L., Advanced Lab in Image Processing, teaching material based on the book Digital Holography and Digital Image Processing: Principles, Methods, Algorithms, Kluwer Academic, 2004.			
3	B. Jahne, Digital Image Processing, Springer, 2002.			
4	Material on the website: http://cs.elfak.ni.ac.rs/nastava/			
5	R. C. Gonzalez and R. E. Woods, Digital Image Processing, 4th ed. Hoboken, NJ: Pearson, 2018.			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, independent work of students on homework and projects, consultations.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	20	Oral exam		50
Colloquia				
Projects	30			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Medical Information Systems		
Lecturer (for lectures)		Janković S. Dragan, Milenković M. Aleksandar		
Lecturer/associate (for exercises)		Milenković M. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The course objectives are introducing students in specificities of medical information system development, deployment and project management. The students will get the knowledge about different types of medical software as well as related standards and legislature.			
Course outcomes	After the course completion, the students will be able to participate in medical information system development process using theoretical and practical knowledge obtained through lectures and software project development.			
Course outline				
Theoretical teaching	Introduction to medical information systems. Data quality . Standards and legislature. Comparison of existing systems . Medical information system design specificities. Aspects of system implementation . System and data security . Overview on implementation technologies. System installation and deployment phase . System exploitation phase. System interoperability. Telemedicine. Decision support systems. Medical information system project management			
Practical teaching (exercises, OFE, study and research work)	Defining main elements of the future information system. Designing database structure and electronic patient record. Medical examination support software components. Patient scheduling software module. Report generation module. Privileges within medical information system. Data security. Designing the data access Web service. Personal health Web portals. Composite electronic health records. Document management systems. Medical information system taxonomy. Data exchange standards. Data synchronization techniques			
Textbooks/references				
	1	Alfred Winter, Elske Ammenwerth, Reinhold Haux, Michael Marschollek, Bianca Steiner, Franziska Jahn, Health Information Systems: Technological and Management Perspectives, Springer, 202		
	2	Joseph Tan, Fay Cobb Payton: Adaptive Health Management Information Systems: Concepts, Cases, and Practical Applications, Third Edition, Jones & Bartlett Publishers; 3 edition (May 21, 2009), ISBN-10: 0763756911		
	3	Dean F. Sittig, Joan S. Ash: Clinical Information Systems: Overcoming Adverse Consequences (Jones and Bartlett Series in Biomedical Informatics), ones & Bartlett Publishers; 1 edition (November 23, 2009), ISBN-10: 0763757640		
	4	Scott Coplan, David Masuda: Project Management for Healthcare Information Technology, McGraw-Hill Professional; 1 edition (February 1, 2011), ISBN-10: 0071740538		
	5	Jingsong Li, Yu Tian, Tianshu Zhou, Healthcare Information Systems: Progress, Challenges and Future Directions, Springer Singapore, 2024.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, Auditive exercises, Laboratory exercises. Student project realization.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		30
Exercises		Oral exam		20
Colloquia				
Projects	50			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Advanced Learning Technologies		
Lecturer (for lectures)		Milentijević Z. Ivan		
Lecturer/associate (for exercises)		Vojinović M. Oliver		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	To enable students to acquire knowledge about contemporary approaches to learning and education, the potential for enhancing learning through information technologies, and the pedagogical and cognitive effects of technology integration in learning.			
Course outcomes	Upon successful completion of the course, students will be able to analyze specific educational objectives and environments, and design technological solutions to support learning. They will gain practical experience in selecting, customizing, and developing components and systems for learning support.			
Course outline				
Theoretical teaching	Learning paradigms – instructivist and constructivist approaches. Instructional design and the use of technological tools in instructional design. Formal and informal learning environments: games, e-learning, computer-supported collaborative learning, problem- and project-based learning, and social interaction-based learning. Computer-assisted testing – generic, adaptive, and standardized testing. Standard formats for presenting learning materials and curricula. Supporting learning with AI. Advanced learning analytics with models.			
Practical teaching (exercises, OFE, study and research work)	Analysis of learning approaches, educational objectives, and learning environments, along with mapping appropriate technologies. Design and development of software tools to support learning.			
Textbooks/references				
	1	Sanju Saha, Santoshi Halder, The Routledge Handbook of Education Technology, Taylor&Francis, 2023.		
	2	Yang Shen, Xin Yin, Yu Jiang, Lingkai Kong, Sheng Li, Haijun Zeng, Case Studies of Information Technology Application in Education: Utilising the Internet, Big Data, Artificial Intelligence, and Cloud in Challenging Times, Springer, 2023.		
	3	Jim Goodell, Learning Engineering Toolkit: Evidence-Based Practices from the Learning Sciences, Instructional Design, and Beyond, Routledge, 2022.		
	4	A curated collection of selected accademic articles		
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, consultations, students' work on assignments and projects, seminars (presentations of student projects, with discussion).			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		10	Written exam	
Exercises		10	Oral exam	40
Colloquia				
Projects		40		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Data Mining Techniques and Methods		
Lecturer (for lectures)		Stojković R. Suzana		
Lecturer/associate (for exercises)		Marković M. Ivica		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The goal of this course is to introduce students to major data mining tasks and with special emphasis on the using the data mining techniques and methods into text analysis and information retrieval systems.			
Course outcomes	After completing this course, students should acquire theoretical knowledge of the principles of the work of the data mining tools and to be able to use existing data mining open source tools and to develop new.			
Course outline				
Theoretical teaching	Data mining goal and application. Types of datasets and attributes. Data quality problems. Data preprocessing. Similarity and desimilarity scores. Exploratory data analysis, summary statistics, data visualization. Classification of data mining algorithms. Linear regression. Classification algorithms: Naive Bayes algorithm, logistic regresion, SVM. Classifier evaluation. Clustering algorrithms: K-means, hierarchical clausterization. Time series. Characteristics of time series: stationarity, trend, seasonality, cyclicity. Linear processes: autoregressive (AR), moving average (MA) processes, and ARMA. Non-stationary time series. Methods for time series modeling, the Box-Jenkins method. Associative analysis. Algorithms for associative rules generation.			
Practical teaching (exercises, OFE, study and research work)	Implementation of all steps of data analysis using open-source libraries: Data preprocessing: outlier detection and removal, handling of missing values, normalization, standardization, dimensionality reduction. Data visualization. Linear regression. Classification algorithms. Evaluation of classification performance. Clustering algorithms. Time series analysis: time series preprocessing, stationarity tests, calculation of correlation and autocorrelation, ARIMA model. Creation of association rules.			
Textbooks/references				
1	Pang-Ning Tan, Anuj Karpatne, Michael Steinbach, Vipin Kumar: Introduction to data mining, 2nd Edition, Addison Wesley, 2019.			
2	D. T. Larose: "Data mining methods and models", JONN WILEY & SONS, 2006			
3	George E. P. Box, Gwilym M. Jenkins, Gregory C. Reinsel, Greta M. Ljung: Time Series Analysis: Forecasting and Constrol, 5th Edition, Wiley, 2015			
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, practical demonstrations, consultations, seminars and projects presentation			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		20	Oral exam	40
Colloquia				
Projects		40		

Specification for the book of courses

Study program	Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering
Type and level of studies	Master Academic Studies
The name of the course	Big Data Systems
Lecturer (for lectures)	Stojanović H. Dragan, Stojanović M. Natalija
Lecturer/associate (for exercises)	Stojanović M. Natalija
Lecturer/associate (for OFE)	
Number of ECTS	6
	Course status (obligatory/elective) Elective
Prerequisites	Databases, Distributed systems
Course objectives	Mastering the knowledge, methods, and technologies necessary for the development of systems and applications for processing and analyzing large volumes of data (Big Data).
Course outcomes	Upon successful completion of the course, the student understands the fundamental concepts, architectures, and technologies for collecting, storing, processing, and analyzing large volumes of data. The student is able to design and implement distributed systems for Big Data processing using appropriate databases, distributed storage systems, and data processing platforms. The student is capable of applying technologies for processing stored data, large-scale data streams, and complex events in real or near-real time. The student is able to develop and apply Big Data systems in cloud, fog, and edge environments, as well as visualize and interpret analytical results. Through practical work, the student is able to independently analyze, implement, evaluate, and optimize Big Data applications in various real-world domains.
Course outline	
Theoretical teaching	Introduction to Big Data Processing and Analytics Systems. Characteristics, types, and formats of big data. Architecture and core components of Big Data systems. Methods and technologies for collecting and storing big data. NoSQL/NewSQL distributed databases (Cassandra, MongoDB, Redis, Druid, Spanner, etc.). Distributed file systems (GFS, HDFS), data lakes, and data lakehouse architectures. Methods and technologies for ingesting and streaming big data (Kafka, Pulsar, NiFi, Flume, etc.). Distributed systems and platforms for managing, processing, and analyzing large stored data (Hadoop, Spark, Flink, etc.). Systems for processing and analyzing fast and high-volume data streams and complex events (Spark Streaming, Flink, etc.). Visualization and visual analytics of big data (InfluxDB, Grafana, etc.). Big data management and analysis in distributed infrastructures across the cloud–fog–edge computing continuum. Analysis, design, implementation, and evaluation of systems and applications for big data processing and analytics. Modern applications based on big data processing and analysis in smart cities, healthcare, industry, transportation, agriculture, traffic systems, as well as social networks/media, and more.
Practical teaching (exercises, OFE, study and research work)	amiliarization with the characteristics, types, and formats of big data and conversion between them. Design and architecture of systems for big data processing and analytics, and implementation technologies. Implementation of methods and technologies for collecting and storing big data. Practical work on storing big data in NoSQL/NewSQL distributed databases: Cassandra, MongoDB, Redis, Druid, Spanner, etc. Storing big data in distributed file systems (GFS, HDFS), data lakes, and data lakehouse architectures. Application and implementation of methods and technologies for big data ingestion and transmission: Kafka, Pulsar, NiFi, etc. Implementation of a software system for big data processing on the Hadoop platform. Implementation of a software system for managing, processing, and analyzing stored big data using the Spark platform and relevant software libraries. Implementation of a distributed system for processing and analyzing big data streams using Spark Streaming. Implementation of a distributed system for processing and analyzing fast and large data streams and complex events using the Flink platform. Implementation of visualization of big data processing and analysis results using InfluxDB, Grafana, and Python libraries. Implementation of a distributed system for big data analysis within the distributed infrastructure of the computing continuum. Implementation of applications based on big data processing and analysis in smart domains: smart cities, healthcare, industry, transportation, agriculture, traffic systems, as well as social networks/media, etc., using publicly available or simulated datasets.
Textbooks/references	

	1	Nathan Marz, James Warren: Big Data Principles and best practices of scalable realtime data systems. Manning Publications Co., 2015.			
	2	Martin Kleppmann, Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems, O'Reilly Media, 2017.			
	3	Jules S. Damji, Brooke Wenig, Tathagata Das, and Denny Lee, Learning Spark: Lightning-Fast Data Analytics, 2nd edition, O'Reilly Media, 2020			
	4	Fabian Hueske and Vasiliki Kalavri, Stream Processing with Apache Flink, O'Reilly Media, 2019.			
	5	Adi Polak, Scaling Machine Learning with Spark: Distributed ML with MLlib, TensorFlow, and PyTorch, O'Reilly Media, 2023			
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods	Lectures, exercises, independent student work on assignments and projects, student seminars.				
Grade (maximum number of points 100)					
Pre-exam duties		Points	Final exam		Points
Activity during lectures			Written exam		
Exercises			Oral exam		40
Colloquia		40			
Projects		20			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Web Mining		
Lecturer (for lectures)		Stojković R. Suzana, Bogdanović D. Miloš		
Lecturer/associate (for exercises)		Bogdanović D. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Web technologies basics, Data structures and algorithms, Databases, Probability and statistics			
Course objectives	Within this course, students will learn the specifics of web document analysis and unstructured data analysis. They should also become familiar with the specific possibilities of applying web mining technologies.			
Course outcomes	Students will acquire theoretical and practical knowledge in the field of document content analysis, classification and clustering of web documents. They should also learn basic techniques and tools for web structure analysis and web access analysis.			
Course outline				
Theoretical teaching	Introduction to Web mining. Web document specifics. Techniques of Web documents content analysis. Document topic identification and ranking of terms in a document. Classification and clustering of documents. Sentiment analysis. Analysis of web structure, ranking documents by importance. Techniques for improving website ratings. Web access analysis. Discovery of patterns of website user behavior. Recommendation systems. Web visualization.			
Practical teaching (exercises, OFE, study and research work)	Introduction to basic Web mining tools and their application. Document classification algorithms, methodologies and tools. Log file structure and log analysis algorithms and tools. Website rating tools. Libraries for implementing machine learning algorithms used in Web mining. Libraries and tools for data visualization. Model training environments used in Web mining.			
Textbooks/references				
	1	Bing Liu, Web Data Mining-Exploring Hyperlinks, Contents, and Usage Data, Second Edition, July 2011, Springer		
	2	Wouter de Nooy, Andrej Mrvar, Vladimir Batagelj, Exploratory Social Network Analysis with Pajek, Cambridge University Press, 2011		
	3	Ulrich Matter, An Introduction to Web Mining: with Applications in R, Springer, 2025, ISBN 978-3-031-96637-8		
	4	Julian McAuley, Personalized Machine Learning, Cambridge University Press, 2022, ISBN 9781009003971		
	5	William L. Hamilton, Graph Representation Learning, Morgan & Claypool/Springer, 2020.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures and demonstration exercises using presentations and interactive work on the computer. Seminar papers and project.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	40
Colloquia				
Projects		60		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Pattern Recognition		
Lecturer (for lectures)		Vučković V. Vladan, Radmanović M. Miloš		
Lecturer/associate (for exercises)		Vučković V. Vladan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The aim of the course is to introduce students to basic methods and techniques for pattern recognition and to acquire sufficient knowledge for independent research work in this field and the practical implementation of pattern recognition applications.			
Course outcomes	In this course, students will acquire knowledge about methods and techniques in the field of pattern recognition that will enable them to successfully solve problems and conduct independent research in the field of pattern recognition.			
Course outline				
Theoretical teaching	Pattern recognition systems, registration and measurement of object characteristics. Data acquisition, preprocessing and signal representation. Complexity reduction and classification. Statistical methods for pattern recognition. Classification based on minimum error. Classification based on maximum similarity. EM-algorithm. SVM (Support Vector Machine) classifiers. Stochastic finite state machines and discrete Markov models. Hidden Markov models. Real-time visual recognition systems. Systems for processing and recognizing digital images and scanned documents. OCR systems. Examples of pattern recognition applications.			
Practical teaching (exercises, OFE, study and research work)	Study research work: Preparation of seminar papers in the field of pattern recognition and oral presentation and defense of papers.			
Textbooks/references				
1	S. Pal, A. Pal, Pattern Recognition From Classical to Modern Approaches, World Scientific, 2001.			
2	S. Bow, Pattern Recognition and Image Preprocessing, CRC Press, 2002.			
3	R. Duda, P. Hart, D. Stork, Pattern Classification, Wiley, 2000.			
4	Teaching material on the website: http://cs.elfak.ni.ac.rs/nastava/			
5	D. Dodou, Pattern Recognition and Classification: An Introduction, 2020, 2nd ed. Delft University of Technology Press.			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures and demonstration exercises using slides and demo examples. Independent research work by students, preparation, presentation and defense of seminar papers. Implementation of projects.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	20	Oral exam		50
Colloquia				
Projects	30			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Data Visualization		
Lecturer (for lectures)		Janković S. Dragan		
Lecturer/associate (for exercises)		Rajković J. Petar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Introducing students to the principles underlying data and information visualization in accordance with the purpose and field. Enabling students to use different methods for data visualization. Introducing existing tools and environments for visualizing data.		
Course outcomes		Students should be introduced to data representation methods and areas of their application. Upon completion of the course, students should be able to select an adequate visual representation for a specific problem, which they can implement in one of the existing tools or develop their own display.		
Course outline				
Theoretical teaching		Data types. Principles of data visualization. Perception. Optical illusions. Tables, charts, trees, graphs, treemaps. Multivariate data visualization. Animations, multimedia presentations. Social network visualization. Medical data visualization, Economic data visualization. Time series. GIS data visualization. Process visualization. Visualization tools and environments (Excel, Tableau, R). Data visualization and analysis. BI environments.		
Practical teaching (exercises, OFE, study and research work)		Practical teaching of visualization of various types of data using: Excel, Tableau, D3.js. Integration of visual data displays into software solutions. Software implementation of individual data visualization algorithms.		
Textbooks/references				
1		Tamara Munzner, Visualization Analysis and Design, CRC Press, 2014.		
2		Scott Murray, "Interactive Data Visualization for the Web", O'Reilly Media, 2013		
3		Claus O. Wilke, "Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures", O'Reilly Media, 2019.		
4		Jonathna Schwabish, "Better Data Visualizations: A Guide for Scholars, Researchers, and Wonks", Columbia University Press, 2021.		
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures, Exercises, Laboratory Exercises		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		20	Oral exam	40
Colloquia				
Projects		40		

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Methods of optimization		
Lecturer (for lectures)			Marinković D. Slađana, Džunić S. Jovana		
Lecturer/associate (for exercises)			Damnjanović G. Ivan		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Obligatory / Elective
Prerequisites	None				
Course objectives	Mastering basic theoretical and practical knowledge in the field of optimization and getting acquainted with various optimization methods. Developing skills of mathematical modelling of real problems of practice, as well as solving them.				
Course outcomes	Students' competence to apply acquired knowledge in the profession. Capability to recognize optimization problems from practice, formulate them precisely, form mathematical models, select appropriate methods for their solution and apply the methods.				
Course outline					
Theoretical teaching	Real multivariable functions. Partial derivatives, gradient, Hessian. Local, constrained and global extreme values. Taylor's formula. Elements of convex analysis. General optimization problem. Linear programming. Duality principle in linear programming. Simplex method. Nonlinear programming. Overview of methods for unconstrained optimization (search methods, gradient methods). Constrained optimizations: necessary and sufficient conditions for optimality. Penalty function methods.				
Practical teaching (exercises, OFE, study and research work)	Solving problems that follow the content presented in the theoretical classes. Special emphasis is placed on looking at the problem as a whole, from recognizing the subject of optimization, through forming a mathematical model and choosing an appropriate method to solving and interpreting the results. Implementation of optimization algorithms through program packages for symbolic calculation. Through the preparation of homework assignments and the analysis of their solutions, the application of mathematical and software tools in the context of optimization is gradually mastered.				
Textbooks/references					
1	E.K.P. Chong, S.H. Zak, An Introduction to Optimization, John Wiley & Sons, 2001.				
2	X.S. Yang, Introduction to Mathematical Optimization, Cambridge International Science Publishing, 2008.				
3	M.J. Kochenderfer, T.A. Wheeler, Algorithms for Optimization, second edition, The MIT Press, 2025.				
4					
5					
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes
2	2	0	0		0
Teaching methods		Lectures, computational exercises, computer exercises, project.			
Grade (maximum number of points 100)					
Pre-exam duties		Points	Final exam		Points
Activity during lectures			Written exam		40
Exercises		10	Oral exam		10
Colloquia					
Projects		40			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Machine Learning		
Lecturer (for lectures)		Stoimenov V. Leonid, Stanimirović S. Aleksandar		
Lecturer/associate (for exercises)		Stanimirović S. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Provide students with knowledge about machine learning, its significance, and its role in the development of intelligent systems for analyzing data from various domains in engineering and science. Enable students to gain insight into the fundamental methods of modern machine learning. Provide information on the ways to implement and apply individual machine learning techniques.			
Course outcomes	Insight into the fundamental methods of modern machine learning. Acquiring knowledge about the role of machine learning in the development of intelligent systems. Theoretical knowledge of the basics of machine learning, including the most important algorithms for supervised and unsupervised learning. Practical application and implementation of machine learning algorithms and/or the use of existing open-source libraries.			
Course outline				
Theoretical teaching	Introduction to machine learning. Approaches and types of machine learning: supervised, unsupervised, and reinforcement learning. The role of machine learning in intelligent systems and big data analysis. Data preparation for machine learning, emphasizing the importance of data quality. Descriptive data analysis. Supervised learning: algorithms and applications in classification and regression. Evaluation and selection of machine learning models. Methods for assessing model performance and criteria for selection. Dimensionality reduction. Techniques for feature reduction, attribute selection, and visualization. Unsupervised learning: discovering patterns in unlabeled data. Reinforcement learning: basic concepts and applications. Artificial neural networks: structure, learning, and applications.			
Practical teaching (exercises, OFE, study and research work)	Practical instruction includes independent or team work by students on identifying real-world problems that can be solved using machine learning techniques. Students design, develop, and evaluate solutions using modern programming languages and tools (Python, Jupyter, Scikit-learn, PyTorch, TensorFlow, Pandas, NumPy, etc.). The focus is on the entire problem-solving process: from data collection and preparation, through selection and application of appropriate algorithms, to model evaluation and result visualization. Practical instruction places special emphasis on: formulating problems from real-world or academic contexts, reasoned algorithm selection, critical analysis of obtained results, and collaborative work and presentation of solutions.			
Textbooks/references				
1	S. Russel, P. Norvig, Artificial intelligence - A Modern Approach, Pearson, 3rd edition, (2016), ISBN-10: 1292153962, ISBN-13: 978-1292153964			
2	G. Hulten, Building Intelligent Systems: A Guide to Machine Learning Engineering, 1st ed. edition (2018), ISBN-10: 1484234316, ISBN-13: 978-1484234310			
3	C. Sammut (Editor), G. I. Webb (Editor), Encyclopedia of Machine Learning and Data Mining, Springer, 2nd ed. (2017), ISBN-10: 148997685X, ISBN-13: 978-1489976857			
4	C.Albon, Machine Learning with Python Cookbook: Practical Solutions from Preprocessing to Deep Learning, O'Reilly Media, (2018), ISBN-13 : 978-1491989388			
5	J. Brownlee, Statistical Methods for Machine Learning: Discover How to Transform Data into Knowledge with Python, Machine Learning Mastery, (2018)			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, whiteboard exercises, lab exercises, independent student work on homework assignments and projects, student seminars (presentations of student work followed by discussions).			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		20	Written exam	
Exercises		20	Oral exam	30

Colloquia			
Projects	30		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Database Management Systems		
Lecturer (for lectures)		Stoimenov V. Leonid, Stanimirović S. Aleksandar		
Lecturer/associate (for exercises)		Stanimirović S. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Mastering knowledge of the basic concepts and principles of database management systems (DBMS) and their components. Gaining knowledge of advanced DBMS techniques (triggers, security, query optimization) and DBMS functions (replication, recovery, backups, fragmentation). Acquiring understanding of the fundamental concepts and principles of operation of advanced database systems, systems for handling large volumes of data, and large distributed systems			
Course outcomes	Theoretical and practical knowledge of database management systems, their components, and methods of use. Theoretical and practical knowledge of advanced DBMS techniques, administration, performance tuning, query optimization, recovery, backups, and data maintenance. Theoretical and practical knowledge of systems for handling large volumes of data and distributed systems supporting social networks and other large-scale systems.			
Course outline				
Theoretical teaching	Introduction to database management systems (DBMS). The role and responsibilities of the database administrator. Database management system: overview of architecture, core modules and functions, examples of such systems. Stored procedures and triggers. Query processing and optimization: concept of query optimization, relational algebra and execution plans, static and dynamic optimization, system catalog, database statistics and optimization. Indexes and index structures, multidimensional indexes, the role of indexes, and recommendations for proper use. Database system security: concept of database system security, security in DBMS, user privileges – granting and revoking (GRANT and REVOKE commands), privilege propagation, view-level security, statistical databases, DAC and MAC security mechanisms. Recovery and transactions. Replication. Backups. Distributed systems and systems for handling large volumes of data. Overview of features in open-source and commercial systems. Overview of DBMS functionalities in object-oriented databases, object-relational DBMS, and NoSQL databases.			
Practical teaching (exercises, OFE, study and research work)	The role of the database administrator, practical use of DBMS and administrative tools – practical examples and tasks. Database performance tuning, creating indexes. Query optimization – practical examples, issues, use of DBMS tools. Security in DBMS and the role of the database administrator, security at the operating system, network, hardware level, etc. – practical examples and tasks. Modern challenges for DBMS – example use of new technologies (object-relational mappers and their use: Hibernate/NHibernate, XML, etc.). NoSQL databases (concept, basic principles, classification, examples) and differences compared to traditional DBMS. Use of systems for big data processing and distributed systems.			
Textbooks/references				
	1	R.Ramakrishnan, J. Gehrke, Database Management Systems,Third Edition, 2002, ISBN-13: 978-0072465631, McGraw-Hill.		
	2	R. Elmasri, S. Navathe, Fundamentals of Database Systems, Pearson; 7 edition (2016), ISBN-13: 9780133970777		
	3	T. M. Conolly, C. E. Begg, Database Systems: A Practical Approach to Design, Implementation and Management, пето издање, 2009		
	4	G. Harrison, Next Generation Databases: NoSQLand Big Data, 1st edition, Apress, 2015, ISBN-13: 978-1484213308		
	5	Literature from the Internet on modern trends in database management systems.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, whiteboard exercises, lab exercises, independent student work on homework assignments and projects, student seminars (presentations of student work followed by discussions).			

Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures		Written exam	
Exercises	70	Oral exam	30
Colloquia			
Projects			

Specification for the book of courses

Study program	Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering
Type and level of studies	Master Academic Studies
The name of the course	Ubiquitous Computing
Lecturer (for lectures)	Stojanović H. Dragan
Lecturer/associate (for exercises)	Predić B. Bratislav
Lecturer/associate (for OFE)	
Number of ECTS	6
Course status (obligatory/elective)	Elective
Prerequisites	Mobile application development, Internet of Things and services
Course objectives	Mastering theoretical and practical knowledge in the field of mobile and ubiquitous computing and the Internet of Things (IoT), as well as methods, technologies, and software tools for the design and implementation of mobile and ubiquitous systems, applications, and services.
Course outcomes	Upon successful completion of the course, the student understands the fundamental concepts, architectures, and technologies of mobile computing, ubiquitous computing, and the Internet of Things. The student is able to design and implement ubiquitous IoT systems that integrate sensors, mobile devices, communication technologies, and services. The student is capable of collecting, transmitting, processing, and analyzing data from the environment, as well as developing location-based and context-aware systems and services. The student understands and applies the principles of AIoT, edge intelligence, and cloud and edge-cloud computing, including the processing of large volumes of data. Through practical work, the student is able to develop and evaluate functional ubiquitous IoT applications while applying appropriate privacy and security mechanisms.
Course outline	
Theoretical teaching	Introduction to Mobile, Ubiquitous Computing, and the Internet of Things (IoT). Modern mobile and ubiquitous computing/communication/sensor systems and devices. Wireless and ad-hoc networks and protocols for synchronous and asynchronous communication. The Internet of Smart and Secure Things; the Web of Things. Sensor and environmental data collection, transmission, and processing. Mobile positioning technologies and location-based systems and applications. Context recognition and the development of context-aware systems and services. The Internet of Smart Things – AIoT (Artificial Intelligence of Things). IoT systems at the edge; intelligence at the edge of infrastructure. IoT systems and services in the cloud and across the edge-cloud computing continuum. Big Data management and analytics in mobile and ubiquitous systems. Privacy and security in mobile, ubiquitous, and IoT systems. Modern mobile/ubiquitous applications and IoT systems in smart cities, homes, vehicles, healthcare, mobility and transportation, industry, agriculture, and more.
Practical teaching (exercises, OFE, study and research work)	Technologies, Components, and Systems of Ubiquitous Computing and IoT. Characteristics and functions of mobile and ubiquitous IoT devices. Communication via synchronous and asynchronous protocols. Application of AI, security, and Web technologies in ubiquitous IoT systems. Methods and technologies for collecting, transmitting, and processing sensor data from the environment. Mobile positioning technologies; development of location-based services (LBS). Implementation of context recognition and application in the development of context-aware IoT systems. Implementation of smart things and the application of AI in AIoT development. Development of IoT systems at the edge infrastructure and implementation of edge intelligence. Development and deployment of IoT systems and services in the cloud and edge-cloud computing continuum. Implementation of ubiquitous IoT systems for big data processing and analytics. Methods and technologies for ensuring privacy and security in IoT systems. Development of ubiquitous applications/IoT systems in current domains: smart cities, homes, vehicles, healthcare, mobility and transportation, industry, agriculture, and more.
Textbooks/references	
1	John Krum (Ed.): Ubiquitous Computing Fundamentals. CRC Press, October 2009.
2	Valentina E. Balas, Vijender Kumar Solanki, Raghvendra Kumar, Manju Khari, Internet of Things and Big Data Analytics for Smart Generation, Springer International Publishing, 2019
3	Perry Lea, IoT and Edge Computing for Architects, 2nd Edition, Packt Publishing, 2020
4	A. King, Programming the Internet of Things: An Introduction to Building Integrated, Device-to-Cloud IoT Solutions, O'Reilly Media, 2021.
5	D. Situnayake, J. Plunkett, AI at the Edge: Solving Real-World Problems with Embedded Machine Learning, O'Reilly Media, 2023.
Number of classes of active education per week during semester/trimester/year	

Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Teaching methods:Lectures, exercises, independent student work on assignments and projects, student seminars.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	40
Colloquia		40		
Projects		20		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering Control Systems / Automatic Control		
Type and level of studies		Master Academic Studies		
The name of the course		Virtual and Augmented Reality Systems		
Lecturer (for lectures)		Milosavljević Lj. Aleksandar, Dimitrijević M. Aleksandar		
Lecturer/associate (for exercises)		Dimitrijević M. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Introducing students to the field of virtual and augmented reality and introducing them to the basic devices, algorithms, and techniques used in the implementation of these systems.		
Course outcomes		Knowledge of the basic principles, devices, techniques and algorithms used in the implementation of virtual and augmented reality systems.		
Course outline				
Theoretical teaching		Introduction to the fields of virtual and augmented reality and historical development. Architecture of virtual reality systems. Input and output devices for virtual reality. Gestural and haptic interface. Geometric modeling of virtual objects. Kinematic and physical modeling. Virtual worlds. Positioning of augmented reality in the continuum between the real world and virtual reality. Classification of augmented reality systems. Techniques for determining the observer's view in augmented reality systems. Augmented virtual environments. Areas and examples of application of virtual and augmented reality systems.		
Practical teaching (exercises, OFE, study and research work)		Practical work includes programming virtual and augmented reality elements in the Unity environment, using the Google Cardboard, Vuforia and AR Foundation libraries. Working with Google Cardboard, students develop simple virtual reality applications for mobile devices, Vuforia is used for augmented reality applications based on image recognition, while AR Foundation enables the creation of more advanced solutions in the field of augmented reality based on surface recognition and interaction with the environment in real time.		
Textbooks/references				
1		K. Stanney, Handbook of Virtual Environments: Design, Implementation, and Applications, Lawrence Erlbaum Associates, 2002.		
2		G. Burdea, P. Coiffet, Virtual Reality Technology, Wiley-IEEE, 2003.		
3		O. Bimber, R. Raskar, Spatial Augmented Reality: Merging Real and Virtual Worlds, A K Peters, 2005.		
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods		Lectures, auditory exercises, laboratory exercises, independent work of students on projects.		
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		30	Oral exam	40
Colloquia				
Projects		30		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering	
Type and level of studies		Master Academic Studies	
The name of the course		DevOps - Integrated Approach to Software Development	
Lecturer (for lectures)		Predić B. Bratislav	
Lecturer/associate (for exercises)		Predić B. Bratislav	
Lecturer/associate (for OFE)			
Number of ECTS	6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None		
Course objectives	The goal of this course is for students to acquire advanced knowledge and skills in the field of DevOps, including process automation, infrastructure as code management, continuous integration and software delivery, monitoring, and security. Through theoretical lectures and practical exercises, students will learn how to improve software development and operational processes, reducing delivery time while increasing the quality and security of applications.		
Course outcomes	Upon successful completion of the course, students will be able to: - Understand the core concepts and principles of the DevOps methodology. - Implement automation processes for software development, testing, and delivery. - Use tools for continuous integration and delivery (CI/CD). - Manage infrastructure using Infrastructure as Code (IaC). - Implement system monitoring and logging. - Optimize application performance and security. - Develop strategies for container orchestration and microservices architectures.		
Course outline			
Theoretical teaching	The following topics provide a balance between theoretical foundations and practical applications, enabling students to gain a comprehensive understanding of DevOps: 1. Introduction to DevOps 2. DevOps Culture and Principles 3. Continuous Integration (CI) 4. Continuous Delivery and Deployment (CD) 5. Version Control and Collaboration 6. Infrastructure as Code (IaC) 7. Containerization and Docker 8. Container Orchestration: Kubernetes 9. Monitoring and Logging 10. Security in DevOps 11. Cloud Computing and DevOps 12. Serverless Architectures and Function as a Service (FaaS) a. Lambda, Azure Functions, Google Cloud Functions b. Benefits and Challenges of Serverless 13. Testing in a DevOps Environment 14. Incident Management and System Resilience 15. DevOps in Practice: Case Study		
Practical teaching (exercises, OFE, study and research work)	The exercises cover key practical aspects of DevOps and prepare students for work in production environments: 1. Working with Git and GitHub – Basic commands, branching, merging, and pull requests. 2. Setting up a Jenkins CI/CD Pipeline – Automating builds, testing, and deployment. 3. Creating Docker Containers – Writing Dockerfiles, building, and running containers. 4. Container Orchestration with Kubernetes – Deployment, scaling, and service management. 5. Using Terraform for Infrastructure Management – Defining cloud infrastructure as code. 6. Implementing Monitoring with Prometheus and Grafana – Collecting and visualizing metrics. 7. Security in DevOps – Working with Vault, container scanning, and secrets management. 8. AWS Lambda and Serverless Functions – Creating and executing serverless functions. 9. Automated Application Testing – Setting up Selenium tests in the CI/CD process. 10. Implementing Logging with the ELK Stack – Configuring and analyzing logs in Elasticsearch.		
Textbooks/references			
1	Kim, G., Behr, K., & Spafford, G. (2013). The Phoenix Project: A Novel About IT, DevOps, and Helping Your Business Win. IT Revolution Press. ISBN: 978-0988262591		

2	Kim, G., Humble, J., Debois, P., & Willis, J. (2016). The DevOps Handbook: How to Create World-Class Agility, Reliability, & Security in Technology Organizations. IT Revolution Press. ISBN: 978-1942788003			
3	Forsgren, N., Humble, J., & Kim, G. (2018). Accelerate: The Science of Lean Software and DevOps: Building and Scaling High Performing Technology Organizations. IT Revolution Press. ISBN: 978-1942788331			
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, auditory exercises, laboratory exercises, and student projects.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	30
Exercises		10	Oral exam	40
Colloquia				
Projects		20		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Management of IT Resources and Services		
Lecturer (for lectures)		Milentijević Z. Ivan		
Lecturer/associate (for exercises)		Milentijević Z. Ivan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	To enable students to acquire knowledge about the role of computer resources and services in organizations, and about ensuring high-quality and reliable IT services.			
Course outcomes	Upon completing the course, students will be able to organize computer resources and services within a specific enterprise/corporation organizational framework, as well as plan the delivery of IT services.			
Course outline				
Theoretical teaching	Management of computer resources; administration methods and systems. Criteria and requirements for computer resources and services within a corporation. Capacity planning: technical infrastructure, service delivery, and budgeting. Capacity and plan control: methods and tools for monitoring technical capacity aspects and performance of networks, servers, workstations, software licenses, information systems, and human resources. Information security: operational procedures, standards, and tools for ensuring data protection. Operational and IT risks: compliance with regulatory requirements. IT control and auditing: service management standards (ITIL, COBIT, ISO, PSP/TSP). Lifecycle management of IT services: service operations, and planning/administration in multi-service environments.			
Practical teaching (exercises, OFE, study and research work)	Collection, organization, and analysis of requirements for computer resources and services. Capacity planning of computer resources: service planning, IT budgeting (capital expenditures, operational costs, and depreciation). Performance testing and control of computer resources: metrics, tools, and methodologies. Implementation examples and characteristics of standard operating procedures; information security testing tools. IT audit execution process: external and internal audits, examples of standards.			
Textbooks/references				
	1	Sutherland, Keith D., and J. Lawrence. A Practical Guide to Service Management: Insights from industry experts for uncovering, implementing, and improving service management practices. Packt Publishing Ltd, 2023.		
	2	Smith, Kevin J. The practical guide to world-class IT service management. Outskirts Press, 2017.		
	3	Maleh, Yassine, and Abdelkebir Sahid, eds. Navigating IT Governance for Resilient Organizations. IGI Global, 2024.		
	4			
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, practical sessions, consultations, and practical project tasks completed individually or in small teams. Presentation and discussion of project solutions.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		10	Written exam	
Exercises		50	Oral exam	40
Colloquia				
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Blockchain Technologies		
Lecturer (for lectures)		Janković S. Dragan, Ćirić M. Vladimir, Stanimirović S. Aleksandar		
Lecturer/associate (for exercises)		Davidović P. Nikola		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		The aim of the course is to introduce students to advanced distributed algorithms and data structures, which enable an understanding of the concept of blockchain systems, as well as to advanced technologies that enable the implementation of these systems.		
Course outcomes		The student is expected to acquire the necessary knowledge to understand the principles and concepts related to blockchain technology, as well as to be able to apply available tools and environments, and implement private and public blockchain systems independently or in available environments. The student is expected to master the technologies in order to be able to independently develop distributed applications on the Ethereum blockchain platform.		
Course outline				
Theoretical teaching		Basic blockchain concepts. Decentralization. Communication in decentralized systems. Elements of cryptography, cryptographic primitives. Public key systems, PKI, RSA, ECC. Hash functions, SHA-1, SHA-2, MD5. Using OpenSSL. Fault tolerance and consensus algorithms. Cryptocurrencies and cryptoeconomics. Public and private blockchain systems. Bitcoin. Digital keys and addresses. Transactions. Mining, CPU, GPU, FPGA, ASIC. Alternative currencies. Ethereum. Smart contracts. Programming in Ethereum. Implementation of the Ethereum system. Development tools. Hyperledger. Implementation of blockchain systems.		
Practical teaching (exercises, OFE, study and research work)		Interacting with a digital wallet. The difference between the main Ethereum network and the test network. Programming in the Solidity programming language. Programming a smart contract. Remix development environment. Basics of the Node.js programming language and the npm package manager. Basics of Javascript ES6 syntax. Truffle tools. Deploying a smart contract on the blockchain. Web3.js and interacting with a smart contract. Testing smart contracts. Basics of React web technology. Creating a web application as a user interface for a distributed application.		
Textbooks/references				
	1	Imran Bashir, "Mastering Blockchain: Distributed ledger technology, decentralization, and smart contracts explained", 2nd Edition, ISBN-10: 978-1788839044, 2018.		
	2	Andreas Antonopoulos, Gavin Wood, "Mastering Ethereum: Building Smart Contracts and DApps", ISBN-10: 978-1491971949, 2019.		
	3	Chris Dannen, "Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners", 1st. edition, ISBN-10: 978-1484225349, 2017.		
	4	Arshdeep Bahga, Vijay Madisetti, "Blockchain Applications: A Hands-On Approach", ISBN-10: 978-0996025553, 2017.		
	5			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, Auditive exercises, Laboratory exercises. Student project realization.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures	10	Written exam		
Exercises	10	Oral exam		40
Colloquia				
Projects	40			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Computer Based Sensor Systems		
Lecturer (for lectures)		Denić B. Dragan, Dinčić R. Milan		
Lecturer/associate (for exercises)		Dinčić R. Milan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	To familiarize students with the fundamental principles, significance, and applications of sensors and sensor systems; techniques for data acquisition, processing, and analysis; modern sensor technologies, including telemetry systems, wireless sensor networks, and Internet of Things (IoT) solutions; programming in LabVIEW; and the development of sensor systems using contemporary hardware-software platforms such as LabVIEW, Raspberry Pi, and FPGA.			
Course outcomes	Students will acquire both theoretical and practical knowledge of modern sensor technologies and systems. Additionally, they will gain skills in hardware and software implementation of sensor systems using LabVIEW, Raspberry Pi, and FPGA platforms.			
Course outline				
Theoretical teaching	Sensors and Sensor Systems. Principles of operation, importance, and applications of sensors. General structure of computer-based sensor systems. Data acquisition. Acquisition cards. Protocols for connecting acquisition cards to a computer. Computer software for data acquisition. LabVIEW software package – features, applications, programming. Concept of virtual instrumentation. Implementation of sensor systems based on LabVIEW, Raspberry Pi, and FPGA platforms. Analysis and processing of measurement data. Telemetry systems. Wireless sensor networks. IoT systems.			
Practical teaching (exercises, OFE, study and research work)	Problem-solving exercises, practical exercises, laboratory work, project assignments, and seminar papers, with the aim of enabling students to acquire practical knowledge related to programming in LabVIEW and to the design and implementation of sensor systems using LabVIEW, Raspberry Pi, and FPGA technologies.			
Textbooks/references				
1	Behzad Ehsani, "Data Acquisition using LabVIEW", Packt Publishing, 2017.			
2	Maurizio Di Paolo Emilio, "Data Acquisition Systems: From Fundamentals to Applied Design", Springer-Verlag, 2013.			
3	Robert Bishop, "Learning With LabVIEW", Pearson, 2015.			
4	National Instruments, "Data Acquisition and Signal Conditioning Course Manual", 2012.			
5	John Shovic, "Raspberry Pi IoT Projects", 2015.			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, problem-solving exercises, laboratory exercises, discussion of student solutions to assigned project tasks, consultations.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures	5	Written exam		25
Exercises	20	Oral exam		25
Colloquia	25			
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Spectral Techniques		
Lecturer (for lectures)		Radmanović M. Miloš		
Lecturer/associate (for exercises)		Radmanović M. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The main goal of the course is for students to acquire theoretical and practical knowledge in the field of application of spectral techniques in the design, analysis and implementation of digital systems.			
Course outcomes	Students will be introduced to various techniques for representing discrete functions, techniques and methods for calculating discrete transformations, and some applications of spectral techniques in the analysis and implementation of digital systems.			
Course outline				
Theoretical teaching	1. Representation of discrete functions, 2. Functional expansions, 3. Decision diagrams, 4. Spectral transformations of logical functions, 5. Techniques and methods for calculating spectral transformations, 6. Fast algorithms, 7. Walsh spectrum, 8. Haar spectrum, 9. Vilenkin–Chrestenson spectrum. 10. Calculation of discrete transformations using decision diagrams, 11. Spectral methods for the synthesis of logic networks. 12. Synthesis of sequential networks using spectral methods, 13. Spectral methods for testing digital systems.			
Practical teaching (exercises, OFE, study and research work)	Work with specialized software packages. Program implementation of algorithms for calculating spectral transformations and their application. Demonstration exercises: 1. Functional developments, 2. Decision diagrams, 3. Spectral transformations of logical functions, 4. Techniques and methods for calculating spectral transformations, 5. Fast algorithms, 6. Walsh spectrum, 7. Haar spectrum, 8. Vilenkin–Chrestenson spectrum. 9. Calculation of discrete transformations using decision diagrams, 10. Spectral methods for synthesizing logic networks. 11. Synthesis of sequential networks using spectral methods, 12. Spectral methods for testing digital systems.			
Textbooks/references				
1	M. Karpovsky, R. Stankovic, J. Astola, Spectral Logic and Its Applications for the Design of Digital Devices, John Wiley & Sons, 2008.			
2	M. A. Thornton, R. Drechsler, D. M. Miller, Spectral Techniques in VLSI CAD, Kluwer Academic Publishers USA, 2012.			
3	R. Drechsler, B. Becker, Decision Diagrams: Theory and Implementation, Springer USA, 2013.			
4	R. Ebendt, G. Fey, and R. Drechsler, Advanced BDD Optimization. Dordrecht, Netherlands, Springer, 2005.			
5	Teaching material on the website: http://cs.elfak.ni.ac.rs/nastava/			
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures and demonstrations using slides, seminars and projects.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises		Oral exam		40
Colloquia				
Projects	60			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Advanced Techniques in 3D Modeling and Animation		
Lecturer (for lectures)		Vučković V. Vladan		
Lecturer/associate (for exercises)		Vučković V. Vladan		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Mastering the basic knowledge necessary for the use of advanced processes and procedures in computer 3D modeling and animation.			
Course outcomes	Theoretical knowledge: Mastering advanced techniques for computer 3D modeling and animation; 3D modeling, programming camera paths, and generating animations on the computer.			
Course outline				
Theoretical teaching	3D Layout - scenes in 3D software. Fundamentals of rendering. Mental Ray – Antialiasing, GI, Final Gather. Mental ray Shaders: Mia X Pass, SSS, Illumination Shaders, Displacement, Ambient Occlusion. Mental Ray – Render Layers, Render Passes, Contribution maps. Hardware render. Lighting. Rendering optimization. Fundamentals of computer animation. Rendering animation. Modeling camera paths. Digital animation direction. Planning and organization of a digitally generated and animated film project. Technical development of the project.			
Practical teaching (exercises, OFE, study and research work)	Exercises; Preparation of seminar papers. Introduction to advanced modeling and animation methods. Practical 3D modeling and animation on the computer. Use of modern 3D animation software. Generating various simple animations.			
Textbooks/references				
1	"Computer Animation, Second Edition: Algorithms and Techniques", Morgan Kaufmann; 2 edition (October 11, 2007); ISBN-10: 0125320000 ISBN-13: 978-0125320009			
2	J. Vince, Introduction to Computer Animation, 4th ed. London: Springer, 2019.			
3	E. Lengyel, Mathematics for 3D Game Programming and Computer Graphics, 3rd ed. Boca Raton, FL: CRC Press, 2019.			
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, blackboard exercises, laboratory exercises, independent student work on homework and projects, consultations.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		30	Oral exam	40
Colloquia		10		
Projects		20		

Specification for the book of courses

Study program			Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies			Master Academic Studies		
The name of the course			Intelligent Information Systems		
Lecturer (for lectures)			Tošić B. Milorad		
Lecturer/associate (for exercises)			Bogdanović D. Miloš		
Lecturer/associate (for OFE)					
Number of ECTS		6	Course status (obligatory/elective)		Obligatory / Elective
Prerequisites		Passed exams in Information Systems, Web Programming, and Artificial Intelligence.			
Course objectives		Acquisition of practical programming skills, theoretical knowledge, and a systematic approach necessary for the design, implementation, and operation of systems in which information technologies, computers, the Internet, and humans collaborate to achieve results characterized as intelligent.			
Course outcomes		Students are able to analyze, design, and implement intelligent information systems in various domains and identify appropriate methods and technologies for their development. They are capable of applying semantic technologies, ontologies, language models, and intelligent agents, as well as integrating and retrieving information from diverse sources. They possess practical skills for developing service-oriented, multi-agent, and language-model-based applications.			
Course outline					
Theoretical teaching		General conceptual assumptions: data models, information, and knowledge; fundamental technologies (Internet, databases, artificial intelligence, information retrieval); business aspects; mathematical foundations. Intelligent databases, deductive and active databases, intelligent information retrieval. Reasoning. Web. The Web and intelligent information systems: Ontologies, Linked Data, Semantic Web, Social Networks, Collaborative Systems, Extensible and Small-World Networks. End-user programming. Architectures of intelligent information systems: service-oriented and multi-agent architectures. Knowledge-based systems. Information retrieval and browsing: Web agents, Web data collection ("crawling"), intelligent Web tools, Web search engines, and the semantics problem. Language models. Development of applications based on language models. Query engineering for language models. Interaction models in systems with intelligent agents based on language models. Data flow models. Development of multi-agent applications. Development of conversational robots (chatbots) on a given topic. Information management, information flow, integration of structured and semi-structured information sources, XML- and RDF-based methods for information representation.			
Practical teaching (exercises, OFE, study and research work)		Tools for ontology development. Programming interfaces for creating RDF knowledge graphs in Java and Python programming languages. SPARQL queries and reasoning in semantic knowledge graphs. Examples in business information systems. Development of prototypes for semantic-enabled intelligent systems. Integration of intelligent systems with tools for automatic data collection from the Web. Tools for summarizing textual and visual data. Application of large language models in intelligent systems. Use of large language models in ontology learning and automation of semantic systems (automatic query generation). Use of multimodal summarization in intelligent systems. Application of Retrieval Augmented Generation (RAG) for information retrieval in intelligent systems. Predictive methods in intelligent information systems. Management of computational infrastructures in intelligent systems – DevOps, DevSecOps, and MLOps tools.			
Textbooks/references					
1		Romero, José Raúl, Inmaculada Medina-Bulo, and Francisco Chicano, eds. Optimising the software development process with artificial intelligence. Springer, 2023.			
2		Huyen, Chip. Designing machine learning systems. " O'Reilly Media, Inc.", 2022.			
3		Phoenix, James, and Mike Taylor. Prompt Engineering for generative AI. " O'Reilly Media, Inc.", 2024.			
4		Fensel, Dieter, Umutcan Simsek, Kevin Angele, Elwin Huaman, Elias Kärle, Oleksandra Panasiuk, Ioan Toma, Jürgen Umbrich, and Alexander Wahler. Knowledge graphs. Switzerland: Springer International Publishing, 2020.			
5		Dietz, Jan LG, and Hans BF Mulder. Enterprise ontology: A human-centric approach to understanding the essence of organisation. Springer Nature, 2024.			
Number of classes of active education per week during semester/trimester/year					
Lectures	Exercises	OFE	Study and research work		Other classes

2	2	0	0	0
Teaching methods	Lectures, exercises, programming assignments			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures		30	Written exam	
Exercises		30	Oral exam	40
Colloquia				
Projects				

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Information Technologies for Development of E-Government Systems		
Lecturer (for lectures)		Stoimenov V. Leonid, Frtunić-Gligorijević B. Milena		
Lecturer/associate (for exercises)		Frtunić-Gligorijević B. Milena		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	Introduction to the basic concepts of e-Government systems and mastery of technologies for designing and implementing various aspects of such systems.			
Course outcomes	Theoretical and practical knowledge of the concepts, design and implementation of basic aspects of e-Government systems			
Course outline				
Theoretical teaching	Introduction to the basic concepts of e-Government systems. A brief overview of the history of e-Government system development. Analysis of key aspects of e-Government: data, e-Services, applications, infrastructure. Data management in e-Government systems: models and databases, data mining techniques, ontologies, semantic analysis. Open data and open data platforms. Analysis of e-Government open data portals. Aspects of information integration: service-oriented architecture, Web services. Development of back-office and front-office applications. Current issues in the development of e-Government infrastructure: cloud technologies. Data storage, storage and management, the concept of Big Data, databases, Data warehouse, Data Lake and other concepts. The place of e-Government in the concept of a Smart City.			
Practical teaching (exercises, OFE, study and research work)	Introduction to open platforms for viewing and storing e-government data. Analysis of open data formats. Open data standards. Semantic analysis and discovery of new knowledge by analyzing existing data. Access to open data portals via API interfaces for data search and analysis. Analysis of data openness according to defined methodologies. Implementation of services for citizens based on available open data.			
Textbooks/references				
	1	Mahmood, Z. (2013). E-Government Implementation and Practice in Developing Countries (pp. 1-348). doi:10.4018/978-1-4666-4090-0		
	2	Reddick, C. & Aikins, Stephen K. (2012). Political, Policy and Management Implications Series: Public Administration and Information Technology (pp. 1- 275), Vol. 1, Springer.		
	3	Vitvar, T., Peristeras, V., & Tarabanis, K. (2010). Semantic Technologies for E-Government. XVI, (pp. 1 - 320), ISBN 978-3-642-03507-4, Springer.		
	4	Garson, G. D. (2006). Public Information Technology and E-Governance: Managing the Virtual State. Boston: Jones & Bartlett. 2006.		
	5	Milić, P., Veljković, N., & Stoimenov, L. (2018). Semantic technologies in e-government: Toward openness and transparency. In Smart Technologies for Smart Governments (pp. 55-66). Springer, Cham.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, auditory exercises, laboratory exercises, independent work of students on homework and projects			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	40	Oral exam		30
Colloquia				
Projects	30			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Fuzzy Logic		
Lecturer (for lectures)		Radmanović M. Miloš		
Lecturer/associate (for exercises)		Radmanović M. Miloš		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites		None		
Course objectives		Students should acquire basic knowledge in the field of Fuzzy Logic and its application in computing, automatic control, and related fields.		
Course outcomes		Upon completion of this course, students should know what Fuzzy systems are, what their characteristics are, how they are described and implemented, and should be able to practically implement simple Fuzzy systems.		
Course outline				
Theoretical teaching		1. Evolution from classical to fuzzy logic, 2. Generating membership functions, 3. T-norms, T-conorms and negations, 4. Fuzzy sets, 5. Logical consistency, 6. Linguistic functions, 7. Probability measures, 8. Fuzzy modeling using “if-then” fuzzy rules, 9. Linguistic modifiers, 10. Aggregation functions, 11. Fuzzy rules, 12. Fuzzy implication analysis and numerical methods for its calculation, 13. Practical applications of fuzzy logic,		
Practical teaching (exercises, OFE, study and research work)		Study research work: seminar papers and independent practical implementation of a simple Fuzzy system. Demonstration exercises: 1. Generating membership functions, 2. T-norms, T-conorms and negations, 3. Fuzzy sets, 4. Logical consistency, 5. Linguistic functions, 6. Probability measures, 7. Fuzzy modeling using “if-then” fuzzy rules, 8. Linguistic modifiers, 9. Aggregation functions, 10. Fuzzy rules. 11. Fuzzy implication analysis and numerical methods for its calculation, 12. Practical applications of fuzzy logic,		
Textbooks/references				
1		M. Mukaidono, Fuzzy Logic for Beginners, World Scientific, 2001.		
2		T. Niimura, T., K. Tanaka, An Introduction to Fuzzy Logic for Practical Applications, Springer New York, 1996.		
3		J. Harris, Fuzzy Logic Applications in Engineering Science, Springer Netherlands, 2011.		
4		Material on the website: Moraga. C., et al. Introduction to Fuzzy Logic, video lectures.		
5		Material on the website: http://cs.elfak.ni.ac.rs/nastava/		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures and demonstrations using slides and interactive computer work. Practical exercises on computers.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	40
Colloquia				
Projects		60		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Artificial Intelligence Applications in Information Systems		
Lecturer (for lectures)		Tošić B. Milorad		
Lecturer/associate (for exercises)		Petrović N. Nenad		
Lecturer/associate (for OFE)				
Number of ECTS	6	Course status (obligatory/elective)	Obligatory / Elective	
Prerequisites	None			
Course objectives	Introducing students to current artificial intelligence techniques and case studies where they can be applied in information systems.			
Course outcomes	Upon completion of the course, the student should be capable of developing services that rely on artificial intelligence, successfully applying AI techniques within information systems, and integrating them effectively.			
Course outline				
Theoretical teaching	Introduction to machine learning. Machine learning in information systems. Supervised machine learning. Unsupervised machine learning. Applications of supervised and unsupervised machine learning in information systems. Introduction to generative artificial intelligence. Large language models. Application of large language models in information systems. Multimodal summarization. Application of multimodal summarization in information systems. Retrieval Augmented Generation (RAG). Knowledge graphs in information systems. Ontologies and semantic technologies. Use of semantics in information systems. Synergy of semantic technologies and large language models. Explainable artificial intelligence. Overview of techniques and applications of explainable AI in information systems. Ontologies and semantic ontologies in the field of explainable artificial intelligence. AI-driven development of information systems.			
Practical teaching (exercises, OFE, study and research work)	Weka API for Java. Machine learning in Python. Examples of classification and regression in information systems. Examples of clustering in information systems. Application of large language models in Python. Application of visual language models in Python. Development of prototypes and integration of LLM services with information systems. RDF representation of semantic knowledge graphs. SPARQL queries. Examples of using semantic graphs in information systems. APIs and libraries for explainable artificial intelligence. Examples of the use of explainable AI in information systems. Application of generative artificial intelligence in the development of information systems.			
Textbooks/references				
1	Alammar J., Grootendorst M., Hands-On Large Language Models, O'Reilly Media, Inc; 2024.			
2	Hitzler P., Rudolph S., Krötzsch M., Foundations of Semantic Web Technologies, Chapman & Hall/CRC, 2011.			
3	Ebert I., Understanding Machine Learning with Java and Weka: A beginner's Guide to Building Predictive Models and Algorithms with Java, 2024.			
4	Rothman D., Hands-On Explainable AI (XAI) with Python: Interpret, visualize, explain, and integrate reliable AI for fair, secure, and trustworthy AI apps, Packt Publishing, 2020.			
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures; Auditory exercises; Laboratory exercises; Computer exercises; Consultations; Independent student research; Oral presentations by students on chosen/assigned topics; Active student participation in classes through the use of the interactive course website.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures	30	Written exam		
Exercises	30	Oral exam		40
Colloquia				
Projects				

Specification for the book of courses

Study program	Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering
Type and level of studies	Master Academic Studies
The name of the course	Medical Signal and Image Processing and Analysis
Lecturer (for lectures)	Janković S. Dragan
Lecturer/associate (for exercises)	Čikić D. Đorđe
Lecturer/associate (for OFE)	
Number of ECTS	6 Course status (obligatory/elective) Obligatory / Elective
Prerequisites	None
Course objectives	Introduction to signals and images encountered in devices used in medicine and methods for analyzing and processing them.
Course outcomes	Complete familiarization with signals and images encountered in medical devices. Mastery of methods for processing and analyzing signals and images encountered in medicine. Developed a critical attitude towards the limitations and benefits of applying signals and images in medicine.
Course outline	
Theoretical teaching	Introduction to medical signal and image processing: Overview of medical signals (EEG, ECG, EMG) and images (RTG, MRI, CT, ultrasound); Basic mathematical models and digital data representation; Sampling and quantization in medical signals and images; Standards and formats of medical signals and images; DICOM Standard, PNG, JPG; Data visualization. Digital medical image processing: Types of signals and systems, LTI systems; Basic signal operations: filtering (Gaussian filter, Median filter, Sharpening), normalization; Fourier transform, Short-time Fourier transform, Spectrogram, Wavelet transform. Digital medical image processing: Intensity transformations of medical images. Spatial transformations of medical images (filtering). Segmentation, Morphological operations on medical images. Fourier transform, Wavelet transform. Sobel filter, Detection of edges, lines and circles. Textures in medical images. Statistical processing of data, signals and images: ICA and PCA for dimensionality reduction; Removing invalid data. Noise filtering in signals and images: mathematical noise models; Gaussian noise, Impulse noise and filtering methods; Periodic and quasi-periodic noise in medical images. Analysis and processing of biomedical signals: Processing and analysis of EEG signals (detection of epileptic seizures). Processing of ECG signals (detection of arrhythmias and analysis of heart rhythm). Processing of EMG signals (recognition of muscle movements). 3D reconstruction and visualization: Basics of 3D medical data (CT and MRI images); Segmentation of organs and tissues in 3D space. Machine learning in medical data processing: Machine learning models for analyzing medical signals. Clustering of medical images (Fuzzy C means clustering, Superpixels...). Deep learning in medical data processing: Application of neural networks in medical data processing. Application of convolutional neural networks for medical image and signal processing. Automated diagnostics: Computer vision in medical diagnostics. Integration of medical data with medical information systems.
Practical teaching (exercises, OFE, study and research work)	Overview of tools: Python (Numpy, SciNumPy, OpenCV, Scikit-learn). Loading and visualization of medical signals and images. Introduction to tools for visualization of signals and images. Applying different filters to signals. Applying Fourier and Wavelet transforms to signals. Calculating signal spectrograms. Filtering (Gaussian filter, Median filter, Sharpening). Otsu, Watershed and Active contours for segmentation. Morphological operations and Morphological reconstruction. Fourier transform. Wavelet transform. Canny edge detector. Segmentation and Hough transform for nucleus detection. Haralick features for textures in medical images. Applying ICA to remove invalid electrodes in EEG signals. Analysis of EEG signal segments. Applying Median filter to filter impulse noise. Spectral filtering of periodic and quasi-periodic noise. Peak detection in ECG signals. EEG signal analysis using Wavelet transform. Data visualization in VTK/ITK libraries. Loading DICOM series and 3D reconstruction of organs. Generating 3D models from CT scans using VTK. Construction of superpixels on the example of histopathological images. Classification of superpixels using machine learning models. Application of Random Forest and SVM for ECG signal analysis. Data processing for migraine prediction. Introduction to TensorFlow and PyTorch libraries. Implementation of CNN models for classification of X-ray images.
Textbooks/references	
1	Rafael C Gonzalez, Richard E Woods, Digital Image Processing, Pearson Education, 4th edition, 2017.

2	Alan Oppenheim, Alan Willsky,S. Hamid, Signals and systems, Pearson, 2015.			
3	Geoff Dougherty, Digital image processing for medical application, Cambridge University Press, 2009.			
4	Kunal Pal, Samir Ari, Arindam Bit, Advanced Methods in Biomedical Signal Processing, Academic Press, 2022.			
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, Auditive exercises, Laboratory exercises.			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises			Oral exam	60
Colloquia				
Projects		40		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Natural Language Processing		
Lecturer (for lectures)		Stojković R. Suzana		
Lecturer/associate (for exercises)		Đorđević D. Nikola		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	None			
Course objectives	The goal of this course is to introduce students to the basic concepts and ideas of the Natural Language Processing (NLP) and with applications of these concepts in information extraction, information retrieval systems, sentiment analysis, question answering systems, and text summarization. Special emphasis is placed on using large language models in these applications.			
Course outcomes	After completing this course the student acquires theoretical and practical knowledge necessary for development of applications based on natural language processing.			
Course outline				
Theoretical teaching	Introduction to NLP. Text tokenizing. Word normalization: stemming and lematization. Type errors correcting, minimum edit distance algorithm. Sentence segmentation. Statistical language models. N-gram model. Part-of-speech tagging. Syntax analysis. CKY algorithm. Semantic analysis. Syntax-directed semantic analysis. Semantic grammars. Text vectorization: TF-IDF weight vectors, word embeddings, sentence embeddings. Large language models. Text classification. Information extraction. Sentiment analysis. Question answering systems, RAG systems. Text summarization.			
Practical teaching (exercises, OFE, study and research work)	Text processing using classical algorithms and open-source libraries such as NLTK: Tokenization. Sentence segmentation. Part-of-speech tagging. Phrase extraction. Stemming and lemmatization. Named entity recognition. Text classification. Sentiment analysis. Text processing using fine-tuning of large language models: Named entity recognition. Text classification. Sentiment analysis. Summarization. Implementation of RAG systems.			
Textbooks/references				
1	D. Jurafsky and J. H. Martin: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition with Language Models, Third Edition, McGraw Hill, 2025.			
2	S. Bird, E. Klein, E.: Natural Language Processing With Python, O'REILLI 2009.			
3				
4				
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, practical demonstrations, consultations, seminars and projects presentation			
Grade (maximum number of points 100)				
Pre-exam duties		Points	Final exam	Points
Activity during lectures			Written exam	
Exercises		20	Oral exam	40
Colloquia				
Projects		40		

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Social Networks Analysis		
Lecturer (for lectures)		Tošić B. Milorad		
Lecturer/associate (for exercises)		Nejković M. Valentina		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Information Systems, Algorithms, Web Programming, Graph Theory, and Artificial Intelligence.			
Course objectives	Acquisition of practical programming skills, theoretical knowledge, and a systematic approach necessary for the design, implementation, and operation of systems in which information technologies, computers, the Internet, and people actively participate in system development.			
Course outcomes	Students are able to model, analyze, and visualize social networks using graph-theoretic methods, ranking algorithms, similarity analysis, and community detection. They are capable of extracting and analyzing social network data and applying machine learning methods for classification, filtering, and recommendation generation. They possess practical skills for implementing and evaluating algorithms on real-world social network data.			
Course outline				
Theoretical teaching	General conceptual assumptions: mathematical foundations, fundamental technologies. Algorithms based on HITS and PageRank methods. Similarity analysis and community detection in social network structures. Information extraction and intelligent reasoning from social networks. Systems for data collection from social media (tag extraction, content extraction, blog search, intelligent web crawling). Intelligent search. Recommendation generation, filtering, and ranking. Feature identification. Advanced classification on social networks. Distributed knowledge in systems with network structure. Visualization of social networks and data analysis. Practical applications and case studies in social network analysis.			
Practical teaching (exercises, OFE, study and research work)	Introduction to the mathematical foundations of social networks – graphs, nodes, edges, adjacency and incidence matrices. Fundamental technologies for social network analysis – data import and processing using tools such as Python (NetworkX) and Gephi. Implementation of the HITS algorithm – calculating authorities and hubs on small network examples. Implementation of the PageRank algorithm – application to web graphs and analysis of ranking convergence. Node similarity analysis – similarity measures and their application in community detection. Community detection in social networks – clustering algorithms. Information extraction from social networks – tag and content analysis using NLP tools. Intelligent search – implementation and analysis of search algorithms optimized for social networks. Recommendation generation – collaborative and content-based filtering, ranking of recommendations. Feature identification and advanced classification – applying machine learning to analyze user behavior. Social network visualization – tools and techniques for visualizing complex network structures. Practical applications and case studies – analysis of real social network data and report preparation.			
Textbooks/references				
	1	Yang, Song, ed. Social Network Analysis in Action: Basic Methods and Applications. Springer, 2025.□		
	2	Zhang, Yanchun, Jeffrey Xu Yu, and Jingyu Hou. Web communities analysis and construction. SpringerVerlag, 2012		
	3	McLevey, John, John Scott, and Peter J. Carrington, eds. The SAGE Handbook of Social Network Analysis. 2nd ed. SAGE, 2023.		
	4	Alag, Satnam. Collective intelligence in action. Manning, 2009.		
	5	Galety, Mohammad Gouse, Chiai Al Atroshi, Buni Balabantaray, and Sachi Nandan Mohanty, eds. Social network analysis: theory and applications. John Wiley & Sons, 2022.		
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0

Teaching methods	Lectures; Classroom exercises; Laboratory exercises; Computer exercises; Consultations; Independent student research; Oral presentations by students on chosen/assigned topics; Active student participation in classes through the use of the interactive course website.		
Grade (maximum number of points 100)			
Pre-exam duties	Points	Final exam	Points
Activity during lectures	30	Written exam	
Exercises	30	Oral exam	40
Colloquia			
Projects			

Specification for the book of courses

Study program		Computer Science and Informatics / Computer Systems Security Computer Science and Informatics / Data Engineering Computer Science and Informatics / Information Systems and Technologies Computer Science and Informatics / Software Engineering		
Type and level of studies		Master Academic Studies		
The name of the course		Deep Learning		
Lecturer (for lectures)		Milosavljević Lj. Aleksandar		
Lecturer/associate (for exercises)		Milosavljević Lj. Aleksandar		
Lecturer/associate (for OFE)				
Number of ECTS		6	Course status (obligatory/elective)	Obligatory / Elective
Prerequisites	Basic knowledge of Python programming, linear algebra, and machine learning fundamentals.			
Course objectives	Introducing students to the field of deep learning. Introduction to basic concepts, training techniques, and architectures of deep neural networks.			
Course outcomes	Knowledge of the basic architectures, techniques, and algorithms used in training deep neural networks. Understanding the advantages of deep learning over traditional approaches. Knowledge of domains where deep learning techniques are applicable and yield good results.			
Course outline				
Theoretical teaching	Introduction to the field of deep learning and historical context. Artificial neuron and artificial neural networks. Neural networks training, error function and parameter optimization. Backpropagation algorithm. Convolutional neural networks. Activation functions, parameter initialization, rejection, batch normalization. Parameter updating during training, network ensembles and voting, data augmentation, transfer learning. Deep learning hardware and software. Convolutional neural network architectures. Visualization and understanding of deep neural networks. Recurrent neural networks. Semantic segmentation, object detection, instance extraction. Generative models, autoencoders, generative adversarial networks. Transformer architecture and large language models. Deep reinforcement learning.			
Practical teaching (exercises, OFE, study and research work)	Introduction to the Python programming language, TensorFlow library, and Keras API. Comparative implementation of training a two-layer neural network using Numpy, basic TensorFlow, and Keras API. Handwritten digit recognition (MNIST dataset) using a layered neural network and a convolutional neural network. Image classification (Caltech-101 dataset) using a pretrained convolutional network and transfer learning. Visualization of the operation of a convolutional neural network for image classification using the Grad-CAM algorithm. Semantic image segmentation (PASCAL VOC dataset) using a U-Net network. Image object detection using a YOLO network. Text sentiment analysis (IMDB dataset) using recurrent neural networks. Text machine translation using an LSTM network. Implementation of a variational autoencoder using the MNIST dataset. Additional training of the transformer network (GPT-2) for text generation using "Serbian folk songs".			
Textbooks/references				
1	Francois Chollet, Deep Learning with Python, 3rd Edition, Manning, 2025.			
2	Howard Huang, Eli Stevens, Luca Antiga, Thomas Viehmann, Deep Learning with PyTorch, 2nd Edition, Manning, 2026.			
3	Ian Goodfellow, Yoshua Bengio, Aaron Courville, Deep Learning, MIT Press, 2016.			
4	Material from the website http://cs231n.stanford.edu/			
5				
Number of classes of active education per week during semester/trimester/year				
Lectures	Exercises	OFE	Study and research work	Other classes
2	2	0	0	0
Teaching methods	Lectures, auditory exercises, independent work of students on projects.			
Grade (maximum number of points 100)				
Pre-exam duties	Points	Final exam		Points
Activity during lectures		Written exam		
Exercises	30	Oral exam		40
Colloquia				
Projects	30			