ŠIAULIAI UNIVERSITY

Informatics (Computer Science)

2020-2021

Course title	E C TS	Degree	Course code	Prerequisites	Subject area
Computer Networks	6	Bachelor	P170B144	Elements of computer's architecture and programming	Computer Science
Software Engineering	6	Bachelor	P175B161	Fundamentals of programming, Object-Oriented programming	Computer Science
Programming Languages	6	Bachelor	P175B157	Programming Fundamentals	Computer Science
Cryptography	3	Bachelor	P170B145	Elements of programming and higher mathematics	Computer Science
Artificial Intelligence	6	Bachelor	P176B001	Accomplishment of modules, related to programming fundamentals, probabilities, algorithm theory, discrete mathematics, graph theory, is necessary	Computer Science
Internet Technologies	3	Bachelor	P175B163	School Information Technology course.	Computer Science
Component – Based Programming	3	Bachelor	P175B125	Programming, Object-oriented programming	Computer Science
Programming for Smart Devices	3	Bachelor	P175B211	Programming, Object-oriented programming	Computer Science
Network Programming	3	Bachelor	P175B162	Elements of computer's architecture, Programming, computer networks	Computer Science
Programmed Control of Servers	3	Bachelor	P170B014	Elements of computer's architecture	Computer Science

Subject area: **Computer Science**

Status	Course code: P170B144 Course title: **COMPUTER NETWORKS** Taught by: Assoc. pr. dr. **Liudvikas Kaklauskas**		
Semester	**ECTS credits**	**Languages**	**Duration**
Autumn or spring	6	Lithuanian,English, Russian	1 semester
Study hours	**Assessment**	**Prerequisites**	**Examination**
Lectures – 32 h Seminars – 0 h Homework – 32 h Self-study – 96 h	10-point scale	Elements of computer's architecture and programming	Mid-term examination – 30% Seminars – 0% Homework – 50% Final examination – 20%

Subject content	Evolution of computer networks. Classification of computer networks, architectures, structure, standards. OSI model. Hardwired and logical tools and services for local and remote communication. Protocols and their sets. TCP/IP. Development of Web applications. Error indication, data compression. Addressing routes and inter-network communication. Security of networks. Internet technologies and services. Network control, domains. Multi-environment networks. Radio, mobile and other modern networks.
Learning Outcomes	Students should acquire knowledge about various structures of computer networks, network hardware and software and their usage opportunities. Also they should be able to choose appropriate tools for design and development of the local (LAN) and global (WAN) computer networks. Students should learn to manage computer network more effectively, use network protocols.

Subject area: **Computer Science**

Status	Course code: P176B001 Course title**: ARTIFICIAL INTELLIGENCE** Taught by: Assoc. prof. dr. **Gražvydas Felinskas**		
Semester	**ECTS credits**	**Languages**	**Duration**
Autumn or Spring	6	Lithuanian, English, Russian	1 semester
Study hours	**Assessment**	**Prerequisites**	**Examination**
Lectures – 32 h Seminars – 0 h Homework – 32 h Self-study – 96 h	*10-point scale*	*Accomplishment of modules, related to programming fundamentals, probabilities, algorithm theory, discrete mathematics, graph theory, is necessary.*	*Mid-term examination – 30%* *Homework, Reporting for laboratory work – 40%* *Final examination – 30%*

Subject content	Notion of artificial intelligence, history, philosophical aspects, components of artificial intelligence. Theoretical branches of artificial intelligence. Agents, communicating agents. The review of search mechanisms, restrictions. Knowledge representation and knowledge bases. Knowledge-based systems, data mining. Reasoning chains, decision trees, semantic networks and frames. Expert systems. Probabilistic reasoning methods, fuzzy logic, coefficient of confidence. Decision making, multi-criteria decisions, strategies, usefulness. Self-learning systems. Design of artificial intelligence systems. Complex search, simulated annealing and genetic methods. AI applicability scope – recognition theory, natural language analysis; usage in development of modern technologies.
Learning Outcomes	Students should acquire knowledge about various branches of artificial intelligence, its application areas and modern achievements. Also they should be able to choose appropriate tools for design and development of the artificial intelligence-based system, to apply artificial intelligence methods in development of various types of software, to choose appropriate methods for knowledge representation, to model various reasoning and search algorithms with standard programming tools. Students should learn to program in logical programming language PROLOG, to represent the facts and rules by the means of language structures, to form queries, to develop the prototype of experimental system. Students should conceptualise the problem areas in finding solutions for the modern complex practical tasks, the importance of application of various heuristic algorithms.
Literature	1. Nils Nilsson. Principles of artificial intelligence. Springer verlag, 1982. 2. Nils Nilsson. Artificial Intelligence: a new synthesis. Morgan Kaufmann Publishers, 1998. 513 p. 3. Stuart Russell, Peter Norvig. Artificial intelligence: a modern approach. Second edition, Prentice Hall, 2003. 1132 p. http://aima.cs.berkeley.edu. 4. George Luger. Artificial intelligence: structures and strategies for complex problem solving (fifth ed.), Addison-Wesley, 2005. 928 p. http://www.cs.unm.edu/~luger/ 5. Mark Stefik. Introduction to knowledge systems. Morgan Kaufmann Publishers, 1995. 871 p. 6. Michael Wooldridge. An introduction to multiagent systems. John Wiley, UK, 2002. 348 p.

Subject area: Computer Science

Status	Course code: **P175B157** Course title: **PROGRAMMING LANGUAGES** Taught by: Assoc. prof. dr. **Vaidas Giedrimas**		
Semester	**ECTS credits**	**Languages**	**Duration**
Autumn or Spring	6	Lithuanian,English	1 semester

Study hours	Assessment	Prerequisites	Examination
Lectures – 32 h Seminars – 32 h Self-study – 96 h	10-point scale	Programming Fundamentals	Reporting for laboratory work – 30% Homework – 30% Exam – 20% Non-traditional tasks (in Moodle environment) – 20%

Subject content	Conception of programming language. History of programming languages. Syntax and semantic of programming language. BNF. Main elements in programming language (concerning the example of programming language for imperative programming). Classifications of programming languages. Programming paradigms. Code translation. Comparative anglysis of chosen programming languages.
Learning Outcomes	Students will acquire knowledge on classifications of programming languages, their common principles, main language elements: types, objects, names, expressions, functions, parameters, and other constructions. They will able to read and understand source code, to manage programming languages constructions, to compare them and to use for practical purposes.
Literature	1. R.W. Sebesta. Concepts of Programming Languages. 5ed. Addison Wesley, 2001. 2. D. P. Friedman, M. Wand. Essentials of Programming Languages, 3ed. MIT Press, 2008. 3. B. W. Kernighan, D.Ritchie, D. M. Ritchie. The C Programming Language, 2ed. Prentice Hall PTR, 1988. 4. T. Crawford, P.Prinz. C: In a Nutshell. O'Reilly, 2005, 618 p. 5. Material in virtual learning environment MOODLE, prepared by lect. Lina Tankeleviciene.

Subject area: **Computer Science**

Status			
	Course code: P170B145 Course title: **CRYPTOGRAPHY** Taught by: Lect. **Mindaugas Stoncelis**		
Semester	**ECTS credits**	**Languages**	**Duration**
Autumn or Spring	3	Lithuanian, English, Russian	1 semester
Study hours	**Assessment**	**Prerequisites**	**Examination**
Lectures – 16 h Seminars – 16 h Homework – 0 h Self-study – 48 h	*10-point scale*	*Elements of and programming and higher mathematics.*	*Reporting for laboratory work – 50%* *Seminars – 0%* *Homework – 20%* *Final examination – 30%*

Subject content	Basics from elementary numbers theory (divisibility, Euclidian algorithm, finite corps). Basics from complexity theory. Classical cryptosystems. Concept of public key cryptosystem. RSA and other cryptosystems. The problem of discrete time. Algorithms for finding prime numbers and factorising the natural numbers. Digital signatures.
Learning Outcomes	This module covers cryptographic primitives like symmetric and public key encryption schemes, digital signatures (based on RSA); also heir security issues are discussed. Besides that, necessary mathematical background is provided.
Literature	1. A. Buchmann. Introduction to Cryptography, Springer, 2001 2. A. Menezes. Handbook of Applied Cryptography, CRC Press, 2001 (http://www.cacr.math.uwaterloo.ca/hac) 3. Cryptography Pointers (http://www.cs.ut.ee/~helger/crypto/) 4. Ron Rivest's Security Links (http://theory.lcs.mit.edu/rivest/crypto-security.html) 5. David Wagner's Crypto Links (http://www.cs.berkeley.edu/~daw/crypto.html) 6. Cryptography Research, Inc. (http://www.cryptography.com/resources/ 7. International Association for Cryptologic Research (http://www.iacr.org) 8. NIST Cryptographic Toolkit (http://csrc.nist.gov/CryptoToolkit/) 9. AES page (http://csrc.nist.gov/encryption/aes/) 10. National Security Agency (http://www.nsa.gov) 11. Bruce Schneiers news letter CRYPTOGRAM(http://www.counterpane.com/crypto-gram.html)

Subject area: **Computer Science**

Status			
	Course code: **P175B163** Course title: **INTERNET TECHNOLOGIES** Taught by: Lect. **Mindaugas Stoncelis**		
Semester	**ECTS credits**	**Languages**	**Duration**
Autumn or spring	3	Lithuanian,English	1 semester
Study hours	**Assessment**	**Prerequisites**	**Examination**
Lectures – 16 h Seminars – 0 h Homework – 32 h Self-study – 32h	10-point scale	School Information Technology course.	Reporting for laboratory work – 50% Seminars – 0% Homework – 20% Final examination – 30%

Subject content	Hardware and software. Addresses of Internet nodes. Protocols for accessing the Web. Transferring files. Ways of connection to remote system. Search. Planning Web page. Netiquette, maintaining Web pages, copyright. HTML language, structure of HTML document. Main elements. Tables. Graphics in HTML document. Additional symbols. Graphical maps and their usage for linking. Frames, connecting several documents. Design of forms and their usage for communicating with user. HTML editors. Programming in JavaScript ant PHP possibilities. Generating Web pages (Wizard and CMS).
Learning Outcomes	To introduce students with modern internet technologies, protocols, services and most popular tools for development of Web pages. To provide with abilities to store information on internet.
Literature	1. Dunaev C. Intranet - technologiji. Web dbc. Cgi. Cobra 2.0. Netscape. Suite. Borland. Intrabuilder. Java. Javascript livewire. 1997m 2. Meloni, Julie C. PHP, MySQL ir Apache. 2007 3. Patrick Carey, New Perspectives on HTML and CSS, 6th Edition, 2013 4. Bruce Lawson and Remy Sharp, Introducing HTML 5, 2011 5. Rob Larsen, Beginning HTML and CSS, 2013

Subject area: **Computer Science**

Status	Course code: P175B161 Course title: **SOFTWARE ENGINEERING** Taught by: Assoc. prof. dr. **Vaidas Giedrimas**		
Semester	**ECTS credits**	**Languages**	**Duration**
Autumn or spring	6	Lithuanian,English, Russian	1 semester
Study hours	**Assessment**	**Prerequisites**	**Examination**
Lectures – 32 h Seminars – 16 h Homework – 32 h Self-study – 80h	10-point scale	Programming, Object-oriented programming	Mid-term examination – 0% Seminars – 0% Homework – 50% Final examination – 50%

Subject content	Software Engineering concept. SE components, PSP and TSP, Project management. Reuse and Code Generation: API and libraries, Design patterns, Reverse engineering, Code generation. Software life cycles. Domain analysis and conceptual modelling. Requirements engineering Software architecture styles. Software design. Software validation. Testing. Documenting. User support. Agile methods.
Learning Outcomes	**In this course the foundations of the software engineering are presented, having in focus importance of each stage of the software lifecycle. Students acquire knowledge required to big software project management. Students get the skills of the code generation, reverse engineering, software testing and documenting.**
Literature	1. L. A. Maciaszek, B. L. Liong Practical software engineering : a case study approach. Pearson/Addison Wesley. 2005 2. I. Sommerville. Software Engineering. Addison Wesley. 2008
	3. O. Pastor, J.C. Molina Model-driven architecture in practice: a software production environment based on conceptual modeling Springer, 2007 4. K. Beck Extreme Programming Explained: Embrace Change. Addison-Wesley 1999 5. K. Czarnecki Generative Programming: Methods, Tools, and Applications. Addison-Wesley 2000 6. A. Endres, D. Rombach. A Handbook of Software Systems Engineering 2003 7. J. W. Moore The Road Map to Software Engineering: A Standards-Based Guide. Wiley-IEEE Computer 8. Society Pr 2006

Subject area: **Computer Science**

Status	Course code: P175B125 Course title: **COMPONENT-BASED PROGRAMMING** Taught by: Assoc. prof. dr. **Vaidas Giedrimas**		
Semester	**ECTS credits**	**Languages**	**Duration**
Autumn or spring	3	Lithuanian,English, Russian	1 semester
Study hours	**Assessment**	**Prerequisites**	**Examination**
Lectures – 16 h Seminars – 0 h Homework – 16 h Self-study – 48 h	10-point scale	Programming, Object-oriented programming	Mid-term examination – 0% Seminars – 0% Homework – 50% Final examination – 50%

Subject content	The component-oriented paradigm. The lifecycle of component-based software. Component concept. Interface of the component. , .NET, EJB, CORBA /CCM component models. Web-services and RESTfull services. Other component models. The tools of CBSE.
Learning Outcomes	**Students get knowledge about the singularities of component-oriented programming, get familiar with various component models, get skills to design and implement components as well as component-based systems**
Literature	1. I. Crnkovic, M. Larsson Building Reliable Component-based Software Systems. Artech House 2002 2. G. T. Heineman, W. T. Councill Component-based software engineering : putting the pieces together. Addison Wesley 2001 1. J. Löwy. Programming .NET Components. O'Reilly 2005 2. C. Szyperski. Component software. Addison-Wesley, 2002 3. A. Ju , W. Kai Qian. Component-oriented programming. Wiley 2006 4. J. Cheesman, J. Daniels. UML Components: A Simple Process for Specifying Component-Based Software. 2002, Addison-Wesley J.

Subject area: **Computer Science**

Status	Course code: P175B211 Course title: **PROGRAMMING FOR SMART DEVICES** Taught by: Assoc. prof. dr. **Vaidas Giedrimas**		
Semester	**ECTS credits**	**Languages**	**Duration**
Autumn or spring	3	Lithuanian,English, Russian	1 semester
Study hours	**Assessment**	**Prerequisites**	**Examination**
Lectures – 16 h Seminars – 0 h Homework – 16 h Self-study – 48 h	10-point scale	Programming, Object-oriented programming	Mid-term examination – 0% Seminars – 0% Homework – 50% Final examination – 50%

Subject content	The module is designed for anyone wanting to learn how to build applications for most popular smartphones and tablets. At theoretical lectures students acquire knowledge about these devices and their OS architecture. At laboratory work acquires the ability to use specialized for these devices programming and software quality management tools. Doing the individually home work creates applications for specific subject area. The core topics are: smart device concept, its architecture; major smart device operating systems; the programming of Apple devices as well as devices with Android OS fundamentals; data storage means (SQLite, CoreData, iCloud, etc.); the market of smart devices applications, their distribution capabilities..
Learning Outcomes	**Students will know the architecture of Smart devices and its OS and they will be able to develop smart-devices-oriented applications and to choose optimal tools for the development.**
Literature	1. Ali, M. (2010). Advanced iOS 4 Programming: Developing Mobile Applications for Apple iPhone, iPad, and iPod touch. Wiley 2. Deitel, P. J. (2013). Android : how to program. Pearson. 3. Harwani, B.M. (2011). Core Data iOS Essentials. Packt publishing. 4. Rogers, R. et al. (2009). Android Application Development. O'Reilly

Subject area: **Computer Science**

Status	Course code: P175B162 Course title: **NETWORK PROGRAMMING** Taught by: Assoc. prof. dr. **Liudvikas Kaklauskas**		
Semester	**ECTS credits**	**Languages**	**Duration**
Autumn or spring	3	Lithuanian,English, Russian	1 semester
Study hours	**Assessment**	**Prerequisites**	**Examination**
Lectures – 32 h Seminars – 0 h Homework – 32 h Self-study – 96 h	10-point scale	Elements of computer's architecture, Programming, computer networks	Mid-term examination – 30% Seminars – 0% Homework – 50% Final examination – 20%

Subject content	Interfaces of applications and protocols. Client and server software design means and their algorithms. Main types of servers. Single process parallel servers. Multi-protocol TCP and UDP servers. Unifies and effective management of parallel server processes. Parallel client processes.
Learning Outcomes	The student will be able to design simplest programs for work in network. The student will be able to select the required technological solution.

Literature	1. Embedded ethernet and internet complete : designing and programming small devices for networking / Jan Axelson. Madison, WI : Lakeview Research LLC, 2003. XIV, 482 p. 2. Java TM network programming / Elliotte Rusty Harold. Beijing, : O'Reilly, 2000. 731, [2] p 3. W3 Schools. Interactyve: http://www.w3schools.com/. 4. Open Merchant Account Ltd. Network Programming in .NET. Interactyve: http://webtropy.com/ 5. Buyya R., Selvi T., Chu X. Object Oriented Programming with Java: Essentials and Applications, 2009, Tata McGraw-Hill Education

Subject area: **Computer Science**

Status	Course code: P170B014 Course title: **PROGRAMMED CONTROL OF SERVERS** Taught by: Assoc. prof. dr. **Liudvikas Kaklauskas**		
Semester	**ECTS credits**	**Languages**	**Duration**
Autumn or spring	3	Lithuanian,English, Russian	1 semester
Study hours	**Assessment**	**Prerequisites**	**Examination**
Lectures – 32 h Seminars – 0 h Homework – 32 h Self-study – 96 h	10-point scale	Elements of computer's architecture, programming, computer networks	Mid-term examination – 30% Seminars – 0% Homework – 50% Final examination – 20%

Subject content	Interface software control commands integrated into network operation systems. Script development and their application for server control. Control automation tools. Ruby, Perl and other programming language, which are best suitable for control of server processes. Development, running and debugging of scripts. Variable types, arrays and their usage for operating system control. Types of operations, their input and output. Control structures. String processing, working with files and directories. Functions, subroutines, packages, modules. Analysis of user and system data, reports.
Learning Outcomes	To learn how to perform complex management of a server's work using network operating system interface (Bash, Sh, cmd, etc.) commands, and capabilities of modern programming languages (Ruby, Perl, etc.).
Literature	1. The Ruby, A Programmer's Best Friend. Interactyve: https://www.ruby-lang.org/en/ 2. The Perl Programming language. Interactyve: http://www.perl.org/ 3. Jean Ross and Greg Stemp. The Windows PowerShell Owner's Manual: Version 2.0. Microsoft Communications Server UA. Interactyve: http://allunifiedcom.files.wordpress.com/2010/07/powershell_v2_owners_manual.pdf. 4. Mike G. BASH Programming - Introduction HOW-TO. Interactyve: http://tldp.org/HOWTO/Bash-Prog-Intro-HOWTO.html. n. 5. Rockwood B. The Cuddletech Guide to SNMP Programming. Interactyve: http://cuddletech.com/articles/snmp/snmp_paper.pdf.