**STUDIJŲ DALYKO (MODULIO) APRAŠAS**

| Dalyko (modulio) pavadinimas | Kodas |
|---|---|
| **Objektinis programavimas** | |

| Dėstytojas (-ai) | Padalinys (-iai) |
|---|---|
| **Koordinuojantis:** dr. Aleksandr Igumenov<br>**Kitas (-i):** prof. dr. Remigijus Paulavičius | Matematikos ir informatikos fakultetas<br>Duomenų mokslo ir skaitmeninių technologijų institutas |

| Studijų pakopa | Dalyko (modulio) tipas |
|---|---|
| Pirmoji | Privalomas |

| Įgyvendinimo forma | Vykdymo laikotarpis | Vykdymo kalba (-os) |
|---|---|---|
| Auditorinė | 2 semestras | Lietuvių |

| Reikalavimai studijuojančiajam | |
|---|---|
| **Išankstiniai reikalavimai:** | **Gretutiniai reikalavimai (jei yra):** |
| Procedūrinis programavimas, Informacinės ir grupinio darbo sistemos | Algoritmai ir duomenų struktūros |

| Dalyko (modulio) apimtis kreditais | Visas studento darbo krūvis | Kontaktinio darbo valandos | Savarankiško darbo valandos |
|---|---|---|---|
| 10 | 271 | 96 | 175 |

| Dalyko (modulio) tikslas: studijų programos ugdomos kompetencijos | | |
|---|---|---|
| Dalyko tikslas – siekiama, kad studentai susipažintų su objektinio programavimo (OP) koncepcija ir ugdytų praktinius gebėjimus kurti efektyvias objektiškai orientuotas programas. | | |
| **Dalyko (modulio) studijų siekiniai** | **Studijų metodai** | **Vertinimo metodai** |
| Gebės naudoti kolektyvinio ir kodo versijavimo sistemas atliekant objektinio programavimo užduotis. | Probleminis dėstymas, diskusija, pavyzdžių analizė, literatūros analizė, laboratorinių užduočių atlikimas. | Laboratorinių darbų atlikimas bei rezultatų gynimas, egzaminas raštu (atvirojo, pusiau atvirojo bei uždarojo tipo klausimai ir užduotys). |
| Gebės paaiškinti objektinio programavimo sąvokas, išmanys jų taikymo sritis. | | |
| Gebės įvertinti objektiškai orientuotos programinės įrangos kūrimo tikslus, taikyti automatinius programinio kodo dokumentavimo įrankius. | | |
| Gebės taikyti pagrindinius objektinio programavimo architektūrinio stiliaus principus užduočių sprendime. | | |
| Gebės parinkti efektyvius algoritmus ir į spartą orientuotas (priklausomai nuo uždavinio specifikos) duomenų struktūras, pritaikyti objektiškai orientuotų sistemų projektavimo ir įgyvendinimo žinias sprendžiant užduotis. | | |
| Gebės naudoti daugiagijinėms sistemoms pritaikytus programinės įrangos derinimo metodus. | | |
| Gebės kurti objektiškai orientuotus programinius sprendimus, pritaikytus didžiųjų duomenų analitikos uždaviniams spręsti. | | |

| Temos | Kontaktinio darbo valandos | | | | | | | Savarankiškų studijų laikas ir užduotys | |
|---|---|---|---|---|---|---|---|---|---|
| | Paskaitos | Konsultacijos | Seminarai | Pratybos | Laboratoriniai darbai | Praktika | **Visas kontaktinis darbas** | **Savarankiškas darbas** | **Užduotys** |
| 1. Kurso apžvalga, C++ standartai, kas yra objektiškai orientuotas programavimas (OOP)?, objekto koncepcija, pažintis su programavimo aplinka: kompiliatorių ir įrankių apžvalga, versijų kontrolės sistemos (git), make/cmake įrankiai, „unit“-testai. | 6 | | | | 2 | | **8** | **16** | Literatūros analizė, laboratoriniai darbai, praktikavimas sprendžiant užduotis: https://leetcode.com/ |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 2. Baziniai duomenų tipai, tipų transformacija, l-reikšmės ir r-reikšmės, rodyklės ir nuorodos, konstantinės nuorodos, dinaminis atminties valdymas, aritmetika su adresais, "išmaniosios" rodyklės, funkcijų persidengimas, direktyvos, įvesties ir išvesties operatoriai. | 8 | | | | 4 | | **12** | **16** | www.topcoder.com https://projecteuler.net/problems |
| 3. Vartotojo tipai, klasės ir objektai, struktūros, konstruktoriai, pagrindinis konstruktorius, destruktoriai, objektinis projektavimas, „RAII" paradigma, inkapsuliavimas, matomumo kontrolė, UML diagramos, dokumentacijos kūrimas su Doxygen. | 8 | | | | 4 | | **12** | **16** | |
| 4. Operatorių persidengimas, įvesties/išvesties operatoriai, operatorių persidengimo realizavimo strategijos, kopijavimo konstruktorius, priskyrimo ir kopijavimo konstruktoriaus palyginimas, seklus ir gilus kopijavimas, nuorodos į r-reikšmes, objektų perkėlimo semantika. | 8 | | | | 4 | | **12** | **16** | |
| 5. Kompozicija ir agregavimas, paveldėjimas, paveldėjimo kontrolė, konstruktoriai ir paveldėjimas, polimorfizmas, virtualiosios funkcijos, ankstyvas ir vėlyvas saistymas. | 8 | | | | 3 | | **11** | **16** | |
| 6. Standartiniai išvesties ir įvesties srautai, failų srautai. | 6 | | | | 3 | | **9** | **16** | |
| 7. Klaidų ir išimčių valdymas, laiko matavimas (std::chrono biblioteka), programos spartos matavimo įrankiai, atsitiktinių skaičių generavimas. | 6 | | | | 4 | | **10** | **16** | |
| 8. Objektiškai orientuotas dizainas, interfeisai, bendrinis programavimas, šablono klasės ir funkcijos, dinaminės bibliotekos, platinamieji paketai (setup.msi/exe). | 7 | | | | 4 | | **11** | **16** | |
| 9. Konteineriai (vektorius, dekas, sąrašas, žemėlapiai, maišos lentelės ir t.t.), iteratoriai. Algoritmų apžvalga: efektyvumas su algoritmais, sparta su duomenų struktūromis. Išimtinių situacijų apdorojimas, laiko matavimas, atsitiktinių skaičių generatoriai. | 7 | | | | 4 | | **11** | **16** | |
| 10. Pasiruošimas egzaminui ir egzamino laikymas. | | | | | | | | **31** | Medžiagos kartojimas |
| **Iš viso** | **64** | | | | **32** | | **96** | **175** | |

| Vertinimo strategija | Svoris proc. | Atsiskaitymo laikas | Vertinimo kriterijai |
|---|---|---|---|
| Pirmasis laboratorinis darbas | 20 | Semestro metu | Studentams skiriamos individualios užduotys, apimančios 1-3 temas. Maksimalus įvertinimas už puikiai atliktas užduotis yra 10 balų (atitinkantys 20 % bendrojo svorio). Skiriami papildomi balai (iki 20 % maksimalaus įverčio svorio), jei užduotys atsiskaitomos anksčiau nurodyto termino. Analogiškai, vėluojant atsiskaityti galutinis įvertinimas yra mažinamas (iki 20 % maksimalaus įverčio svorio). |
| Antrasis laboratorinis darbas | 20 | Semestro metu | Studentams skiriamos individualios užduotys, apimančios 4-6 temas. Maksimalus įvertinimas už puikiai atliktas užduotis yra 10 balų (atitinkantys 20% bendrojo svorio). Skiriami papildomi balai (iki 20 % maksimalaus įverčio svorio), jei užduotys atsiskaitomos anksčiau nurodyto termino. Analogiškai, vėluojant atsiskaityti galutinis įvertinimas yra mažinamas (iki 20 % maksimalaus įverčio svorio). |
| Trečiasis laboratorinis darbas | 20 | Semestro metu | Studentams skiriamos individualios užduotys, apimančios 7-9 temas. Maksimalus įvertinimas už puikiai atliktas užduotis yra 10 balų (atitinkantys 20% bendrojo svorio). Skiriami papildomi balai (iki 20 % maksimalaus įverčio svorio), jei užduotys atsiskaitomos anksčiau nurodyto termino. Analogiškai, vėluojant atsiskaityti galutinis įvertinimas yra mažinamas (iki 20 % maksimalaus įverčio svorio). |
| Egzaminas (raštu) | 40 | Egzaminų sesijos metu | Egzaminą laikyti leidžiama semestro metu surinkus ne mažiau 7.5 balų skaičių, atitinkantį 25% laboratoriniams darbams skirtojo svorio. Egzamino metu galima surinkti iki 10 taškų, kurie atitinka 40% galutinio įvertinimo. Egzaminas susideda iš dviejų etapų. Pirmiausia, studentas turi atsakyti į klausimus iš paskaitose pateiktų temų (iki 2 taškų). Antroje egzamino dalyje studentas turi pateikti praktinį pateiktos problemos sprendimą C++ kalboje (iki 8 taškų), motyvuojant naudojamų priemonių efektyvumą bei analizuojant alternatyvius užduoties sprendimo būdus. |

| Autorius | Leidimo metai | Pavadinimas | Periodinio leidinio Nr. ar leidinio tomas | Leidimo vieta ir leidykla ar internetinė nuoroda |
|---|---|---|---|---|
| **Privaloma literatūra** | | | | |
| Bjarne Stroustrup | 2022 | A Tour of C++ | Third Edition | Addison-Wesley Professional; https://www.stroustrup.com/tour3.html |
| Bjarne Stroustrup | 2014 | Programming: Principles and Practice Using C++ | 2nd Edition | Addison-Wesley, http://www.stroustrup.com/programming.html |
| Stanley Lippman, Josée Lajoie, and Barbara E. Moo | 2012 | C++ Primer | 5th Edition | Addison-Wesley, http://www.informit.com/store/c-plus-plus-primer-9780321714114 |
| Bjarne Stroustrup | 2013 | The C++ Programming Language | 4th Edition | Addison-Wesley, http://www.stroustrup.com/4th.html |
| **Papildoma literatūra** | | | | |
| Nicolai Josuttis | 2012 | C++ Standard Library Tutorial and Reference | 2nd Edition | Addison Wesley Longman, http://cppstdlib.com/ |
| Anthony Williams | 2012 | C++ Concurrency in Action: Practical Multithreading | | Manning Publications Co |
| Scott Meyers | 2014 | Effective Modern C++: 42 Specific Ways to Improve Your Use of C++11 and C++14 | | O'Reilly, http://shop.oreilly.com/product/0636920033707.do |
| Scott Meyers | 2005 | Effective C++: 55 Specific Ways to Improve Your Programs and Designs | 3rd Edition | Addison-Wesley, http://www.aristeia.com/books.html |
| Scott Meyers | 2001 | Effective STL: 50 Specific Ways to Improve Your Use of the Standard Template Library | | Addison-Wesley, http://www.aristeia.com/books.html |
| Herb Sutter | 1999 | Exceptional C++: 47 Engineering Puzzles, Programming Problems, and Solutions | | http://www.gotw.ca/publications/xc++.htm |
| Andrew Koenig and Barbara Moo | 2000 | Accelerated C++: Practical Programming by Example | | Addison-Wesley, http://www.informit.com/store/accelerated-c-plus-plus-practical-programming-by-example-9780201703535 |

**COURSE UNIT (MODULE) DESCRIPTION**

| Course unit (module) title | Code |
|---|---|
| **Object Oriented Programming** | |

| Lecturer(s) | Department(s) where the course unit (module) is delivered |
|---|---|
| **Coordinator:** dr. Aleksandr Igumenov<br>**Other(s): ):** prof. dr. Remigijus Paulavičius | Faculty of Mathematics and Informatics<br>Institute of Data Science and Digital Technologies |

| Study cycle | Type of the course unit (module) |
|---|---|
| First | Compulsory |

| Mode of delivery | Period when the course unit (module) is delivered | Language(s) of instruction |
|---|---|---|
| face-to-face | 2nd semester | Lithuanian |

| Requirements for students | |
|---|---|
| **Prerequisites:**<br>Procedural programming, Information and Groupware Systems | **Additional requirements (if any):**<br>Algorithms and Data Structures |

| Course (module) volume in credits | Total student's workload | Contact hours | Self-study hours |
|---|---|---|---|
| 10 | 271 | 96 | 175 |

| Purpose of the course unit (module): programme competences to be developed | | |
|---|---|---|
| The purpose of the course is to give a grounding in the key concepts of object-oriented programming (OOP) and taught to design and effectively implement object-oriented programs. | | |
| **Learning outcomes of the course unit (module)** | **Teaching and learning methods** | **Assessment methods** |
| Will be able to use groupware systems for team work. | Problem oriented teaching, working in a group, group discussion, case studies, individual work, consultations, laboratory works and literature analysis. | Assessment of laboratory works, written exam (open, semi-open and closed questions and tasks). |
| Will be able to program using basic concepts of OOP. | | |
| Will be able to analyse the problem domain, identify needs of system development and update, and prepare Doxygen type documentations. | | |
| Will be able to understand the concept of the object-oriented programming (OOP). | | |
| Will be able to choose efficient algorithms and performance oriented data structures (depending on the problem), to apply object oriented system development knowledge. | | |
| Will be able to design and implement algorithms for multiprocessor and multithreaded systems, run high-performance computing tasks. | | |
| Will be able to plan, schedule and perform experiments, evaluate the results, and draw conclusions. | | |

| Content: breakdown of the topics | Contact hours | | | | | | | Self-study work: time and assignments | |
|---|---|---|---|---|---|---|---|---|---|
| | Lectures | Tutorials | Seminars | Exercises | Laboratory work | Internship/work placement | **Contact hours** | **Self-study hours** | **Assignments** |
| 1. Course overview, C++ standards, what is object-oriented programming (OOP), concept of the object, choosing the right IDE and compiler, version control systems (git), make/cmake tools, unit-testing. | 6 | | | | 2 | | **8** | **16** | Analysis of the literature, laboratory works, practice coding and problem-solving: |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 2. Core language types and type conversions, l-values and r-values, pointers, dynamic allocation, address arithmetic, array pointers, (constant) references, smart pointers, overloaded functions, inline functions, returning by reference, directives, output and input operators, namespaces. | 8 | | | | 4 | | **12** | **16** | https://leetcode.com/ www.topcoder.com https://projecteuler.net/problems |
| 3. User-defined types, classes, struct, objects, constructors, default constructor, destructors, object-oriented design, C++ garbage collector through RAII paradigm, encapsulation, access specifiers, this pointer, static member variables and static member functions, UML diagrams, generating documentation with Doxygen. | 8 | | | | 4 | | **12** | **16** | |
| 4. Operator overloading, overloading the I/O operators, overloading operators using standard and member functions, copy constructor, assignment vs. copy constructor, shallow copy vs. deep copy, ) r-value references, move semantics, “rule of 3” and “rule of 5”. | 8 | | | | 4 | | **12** | **16** | |
| 5. Object composition and aggregation, class inheritance, inheritance and access specifiers, constructors and initialization of derived classes, polymorphism, virtual functions, early and late binding. | 8 | | | | 3 | | **11** | **16** | |
| 6. Standard output and input streams, std::string, file streams. | 6 | | | | 3 | | **9** | **16** | |
| 7. Error and exception handling, measuring performance (std::chrono library), profiling tools, random number generator(s). | 6 | | | | 4 | | **10** | **16** | |
| 8. Object-oriented design and patterns, interface, generic programming, template classes and functions, dynamic link libraries, distribution packages (setup.msi/exe). | 7 | | | | 4 | | **11** | **16** | |
| 9. An overview of containers (vector, deque, set, map, hashtables and etc.), iterators, and algorithms. Efficiency with algorithms, performance with data structures. | 7 | | | | 4 | | **11** | **16** | |
| 10. Preparation for the exam and taking the exam. | | | | | | | | **31** | Literature review |
| **Total** | **64** | | | | **32** | | **96** | **175** | |

| Assessment strategy | Weight,% | Deadline | Assessment criteria |
|---|---|---|---|
| The first laboratory work | 20 | During the semester | Individual works are assigned to students covering topics 1-3. The maximum score for the assignment is 10 points (this corresponds 20% of the total weight). Students can receive bonus points (up to 20% of the maximum score) when tasks are successfully defended before the deadline. Similarly, the final assessment can be reduced (up to 20% of the maximum score) due to delays. |
| The second laboratory work | 20 | During the semester | Individual works are assigned to students covering topics 4-6. The maximum score for the assignment is 10 points (this corresponds 20% of the total weight). Students can receive bonus points (up to 20% of the maximum score) when tasks are successfully defended before the deadline. Similarly, the final assessment can be reduced (up to 20% of the maximum score) due to delays. |
| The third laboratory work | 20 | During the semester | Individual works are assigned to students covering topics 7-9. The maximum score for the assignment is 10 points (this corresponds 20% of the total weight). Students can receive bonus points (up to 20% of the maximum score) when tasks are successfully defended before the deadline. Similarly, the final assessment can be reduced (up to 20% of the maximum score) due to delays. |
| Written examination | 40 | During the exam session | The final exam is allowed to take if the minimum qualifying mark (equal to 7.5 points; equivalently to 25% of the total score from laboratory works). During the exam, students can collect up to 10 points, which corresponds to 40% of the final assessment. The examination consists of two stages. First, the student must answer different complexity questions corresponding topics covered on lectures (student can earn up to 2 points). In the second part, the student must provide a practical solution to the actual problem (real-world situation) (student can earn up to 8 points). Performance analysis, as well as analysis of alternative solution strategies, must be provided. |

| Author | Year of publication | Title | Issue of a periodical or volume of a publication | Publishing place and house or web link |
|---|---|---|---|---|
| **Compulsory reading** | | | | |
| Bjarne Stroustrup | 2022 | A Tour of C++ | Third Edition | Addison-Wesley Professional; https://www.stroustrup.com/tour3.html |
| Bjarne Stroustrup | 2014 | Programming: Principles and Practice Using C++ | 2nd Edition | Addison-Wesley, http://www.stroustrup.com/programming.html |
| Stanley Lippman, Josée Lajoie, and Barbara E. Moo | 2012 | C++ Primer | 5th Edition | Addison-Wesley, http://www.informit.com/store/c-plus-plus-primer-9780321714114 |
| Bjarne Stroustrup | 2013 | The C++ Programming Language | 4th Edition | Addison-Wesley, http://www.stroustrup.com/4th.html |
| **Optional reading** | | | | |
| Nicolai Josuttis | 2012 | C++ Standard Library Tutorial and Reference | 2nd Edition | Addison Wesley Longman, http://cppstdlib.com/ |
| Anthony Williams | 2012 | C++ Concurrency in Action: Practical Multithreading | | Manning Publications Co |
| Scott Meyers | 2014 | Effective Modern C++: 42 Specific Ways to Improve Your Use of C++11 and C++14 | | O'Reilly, http://shop.oreilly.com/product/0636920033707.do |
| Scott Meyers | 2005 | Effective C++: 55 Specific Ways to Improve Your Programs and Designs | 3rd Edition | Addison-Wesley, http://www.aristeia.com/books.html |
| Scott Meyers | 2001 | Effective STL: 50 Specific Ways to Improve Your Use of the Standard Template Library | | Addison-Wesley, http://www.aristeia.com/books.html |
| Herb Sutter | 1999 | Exceptional C++: 47 Engineering Puzzles, Programming Problems, and Solutions | | http://www.gotw.ca/publications/xc++.htm |
| Andrew Koenig and Barbara Moo | 2000 | Accelerated C++: Practical Programming by Example | | Addison-Wesley, http://www.informit.com/store/accelerated-c-plus-plus-practical-programming-by-example-9780201703535 |