STUDIJŲ DALYKO (MODULIO) APRAŠAS

Dalyko (modulio) pavadinimas	Kodas
Projektavimo šablonai	

Dėstytojas (-ai)	Padalinys (-iai)
Koordinuojantis: Justinas Jucevičius **Kitas (-i):**	Matematikos ir informatikos fakultetas Duomenų mokslo ir skaitmeninių technologijų institutas

Studijų pakopa	Dalyko (modulio) tipas
Pirmoji	Pasirenkamas

Įgyvendinimo forma	Vykdymo laikotarpis	Vykdymo kalba (-os)
Auditorinė	7 semestras	Lietuvių / Anglų

Reikalavimai studijuojančiajam	
Išankstiniai reikalavimai: Procedūrinis programavimas, Objektinis programavimas.	**Gretutiniai reikalavimai (jei yra):**

Dalyko (modulio) apimtis kreditais	Visas studento darbo krūvis	Kontaktinio darbo valandos	Savarankiško darbo valandos
5	133	64	69

Dalyko (modulio) tikslas: studijų programos ugdomos kompetencijos		
Dalyko tikslas – siekiama, kad studentai susipažintų su moderniais objektinio projektavimo šablonais, užtikrinančiais sėkmingą objektinės programinės įrangos kūrimą, atnaujinimą ir palaikymą, ugdytų praktinius gebėjimus taikyti objektinio projektavimo šablonus kuriant programinę įrangą		
Dalyko (modulio) studijų siekiniai	**Studijų metodai**	**Vertinimo metodai**
Gebės taikyti tinkamą programinės įrangos kūrimo metodiką, leidžiančią sprendimus grįsti projektavimo šablonais.	Probleminis dėstymas, diskusija, literatūros analizė, pavyzdžių analizė, savarankiškas darbas, laboratoriniai darbai.	Laboratorinių darbų atlikimas bei rezultatų gynimas, egzaminas raštu (atvirojo, pusiau atvirojo bei uždarojo tipo klausimai ir užduotys).
Gebės atlikti programinės įrangos vidinių pranešimų siuntimo analizę ir parinkti tinkamus projektinius sprendimus.		
Gebės projektavimo šablonus panaudoti sprendimus įgyvendinant našiųjų skaičiavimų aplnkoje, parinkti optimalias programines įgyvendinimo priemones.		
Gebės sudaryti projektavimo šablonams skirtus testus ir testuoti daugiagiję programinę įrangą, taikant nuodugnaus sistemų projektavimo metodiką.		
Gebės taikyti atvirkštinės inžinerijos metodus esamiems programiniams sprendimams, projektavimo šablonams modifikuoti, keliems apjungti ir pakeitimams dokumentuoti.		
Gebės kritiškai įvertinti esamų programinių sprendimų pagrįstumą, taikyti struktūrinius, objektus kuriančiuosius ir elgesio šablonus sistemų integracijos klausimams spręsti.		

Temos	Kontaktinio darbo valandos							Savarankiškų studijų laikas ir užduotys	
	Paskaitos	Konsultacijos	Seminarai	Pratybos	Laboratoriniai darbai	Praktika	**Visas kontaktinis darbas**	**Savarankiškas darbas**	**Užduotys**
1. Kas yra projektavimo šablonai? Kaip išsirinkti ir projektuoti juos? Sąsajos ir jų realizavimas, kompozicija ir paveldėjimas, agregavimas.	4				4		**8**	**8**	Literatūros analizė, laboratoriniai darbai, praktikavimas sprendžiant praktines užduotis.
2. Projektavimo šablonų klasifikacija: kuriantieji šablonai, struktūriniai šablonai, elgesio šablonai.	4				4		**8**	**10**	
3. Strateginis klasės projektavimas, šablonų naudojimo privalumai, kompiliavimo tvirtinimai.	4				4		**8**	**8**	

4. Singletonai, singletonų realizacija, unikalumo reikalavimas, išmaniosios rodyklės ir daugiagijiškumas, daugiagijiško kodo testavimas.	4				4		**8**	**10**	
5. Objektų gamyklos: klasės ir objektai, jų realizacija ir apibendrinimas, abstrakčios gamyklos, klonavimo gamyklos, jų panaudojimas su kitais šabloninio programavimo elementais.	4				4		**8**	**8**	
6. Lankytojai, lankytojų realizavimas, multi-metodai: kas jie yra ir kada juos naudoti?	4				4		**8**	**8**	
7. Klaidų tvarkymas, unit testai, nuodugnus sistemos projektavimas etapais.	8				8		**16**	**8**	
Egzaminas								**9**	Literatūros kartojimas
Iš viso	**32**				**32**		**64**	**69**	

Vertinimo strategija	**Svoris proc.**	**Atsiskaitymo laikas**	**Vertinimo kriterijai**
Pirmasis laboratorinis darbas	20	Semestro metu	Studentams skiriamos individualios užduotys, apimančios 1-3 temas. Maksimalus įvertinimas yra 2 balai (atitinkantys 20 % bendrojo svorio). Skiriami papildomi balai (iki 25 % maksimalaus įverčio svorio), jei užduotys atsiskaitomos anksčiau nurodyto termino. Analogiškai, vėluojant atsiskaityti, galutinis įvertinimas yra mažinamas (iki 50 % maksimalaus įverčio svorio).
Antrasis laboratorinis darbas	20	Semestro metu	Studentams skiriamos individualios užduotys, apimančios 4-6 temas. Maksimalus įvertinimas yra 2 balai (atitinkantys 20 % bendrojo svorio). Skiriami papildomi balai (iki 25 % maksimalaus įverčio svorio), jei užduotys atsiskaitomos anksčiau nurodyto termino. Analogiškai, vėluojant atsiskaityti, galutinis įvertinimas yra mažinamas (iki 50 % maksimalaus įverčio svorio).
Egzaminas (raštu)	60	Egzaminų sesijos metu	Egzaminą laikyti leidžiama semestro metu surinkus ne mažiau 2 balų, kurie atitinka 50 % laboratoriniams darbams skirtojo svorio. Egzamino metu galima surinkti iki 6 taškų, kurie atitinka 60 % galutinio įvertinimo. Egzamino vertinimas susideda iš atvirojo, pusiau atvirojo bei uždarojo tipo klausimų ir užduočių. Egzaminas gali būti laikomas eksternu, kai už atliktus laboratorinius darbus surinkta ne mažiau 50 % laboratoriniams darbams skirtojo įverčio.

Autorius	**Leidimo metai**	**Pavadinimas**	**Periodinio leidinio Nr. ar leidinio tomas**	**Leidimo vieta ir leidykla ar internetinė nuoroda**
Privaloma literatūra				
Andrei Alexandrescu	2001	Modern C++ Design: Generic Programming and Design Patterns Applied		Addison-Wesley Professional
Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides	1994	Design Patterns: Elements of Reusable Object-Oriented Software		Addison-Wesley
Robert C. Martin	2008	Clean Code: A Handbook of Agile Software Craftsmanship		Prentice Hall
Papildoma literatūra				
Eric Freeman, Elisabeth Robson, Bert Bates, Kathy Sierra	2004	Head First Design Patterns		O'Reilly Media
David Abrahams, Aleksey Gurtovoy	2004	C++ Template Metaprogramming: Concepts, Tools, and Techniques from Boost and Beyond, Portable Documents		Addison-Wesley Professional

COURSE UNIT (MODULE) DESCRIPTION

Course unit (module) title	Code
Design Patterns	

Lecturer(s)	Department(s) where the course unit (module) is delivered
Coordinator: Justinas Jucevičius **Other(s):**	Faculty of Mathematics and Informatics Institute of Data Science and Digital Technologies

Study cycle	Type of the course unit (module)
First	Optional

Mode of delivery	Period when the course unit (module) is delivered	Language(s) of instruction
face-to-face	7th semester	Lithuanian / English

Requirements for students	
Prerequisites: Procedural programming, Object-Oriented Programming	**Additional requirements (if any):**

Course (module) volume in credits	Total student's workload	Contact hours	Self-study hours
5	133	64	69

Purpose of the course unit (module): programme competences to be developed		
The purpose of the course unit is to teach object-oriented design techniques and features that lead to a good object-oriented software design for real-world applications, develop practical design pattern application skills.		
Learning outcomes of the course unit (module)	**Teaching and learning methods**	**Assessment methods**
Ability to select appropriate software development methodology that enables efficient use of design patterns.	Problem oriented teaching, working in a group, group discussion, individual work, consultations, laboratory works, learning through dedicated web pages.	Assessment of laboratory works, written exam (open, semi-open and closed questions and tasks).
Ability to analyse inner process of the message passing between objects and adopt appropriate behavioural pattern.		
Ability to implement design patterns for high performance computation, select optimal implementation tools.		
Ability to develop and run design pattern testing scripts, apply step-by-step and other error handling and bug fixing techniques.		
Ability to apply reverse or re-engineering methods for the design pattern modification, patterns integration and update documentation.		
Ability to apply behavioural, creational and creational patterns for system integration tasks.		

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. What is a design pattern? How to select and use design pattern? Interfaces and implementations, composition vs. inheritance, aggregation vs. acquaintance.	4				4		**8**	**8**	Analysis of the literature, laboratory works, practice coding and problem-solving.
2. Design pattern classification: creational patterns, structural patterns, behavioral patterns.	4				4		**8**	**10**	
3. Policy-based class design, the benefit of templates, policies and policy classes, compile-time assertions.	4				4		**8**	**8**	

4. Singletons (static data + static functions != singletons) implementing singletons, enforcing uniqueness, the dead reference problem, smart pointers and multithreading, concurrency, concurrency defense principles, testing threaded code.	4				4		**8**	**10**	
5. Object factories: classes and objects, implementing object factories, generalization, clone factories, using object factories with other generic components, abstract factory.	4				4		**8**	**8**	
6. Visitors, generic implementation of the visitor, multi-methods: what they are and when are they needed?	4				4		**8**	**8**	
7. Error handling, unit tests, designing a complete program: step-by-step study.	8				8		**16**	**8**	
8. Preparation for the exam and taking the exam.								**9**	Literature review
Total	**32**				**32**		**64**	**69**	

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 2 points (this corresponds 20% of the total weight). Students can receive bonus points (up to 25% of the maximum score) when tasks are successfully defended before the deadline. Similarly, the final assessment can be reduced (up to 50% 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 2 points (this corresponds 20% of the total weight). Students can receive bonus points (up to 25% of the maximum score) when tasks are successfully defended before the deadline. Similarly, the final assessment can be reduced (up to 50% of the maximum score) due to delays.
Written examination	60	During the exams session	The final exam is allowed to take if the minimum qualifying mark (equal to 2 points; equivalently to 50% of the total score from laboratory works). During the exam, students can collect up to 6 points, which corresponds to 60% of the final assessment. The examination consists of open, semi-open and closed questions and tasks. Taking the final exam externally is allowed if scored at least 50 % of the total score from laboratory works.

Author	**Year of publication**	**Title**	**Issue of a periodical or volume of a publication**	**Publishing place and house or web link**
Compulsory reading				
Andrei Alexandrescu	2001	Modern C++ Design: Generic Programming and Design Patterns Applied		Addison-Wesley Professional
Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides	1994	Design Patterns: Elements of Reusable Object-Oriented Software		Addison-Wesley
Robert C. Martin	2008	Clean Code: A Handbook of Agile Software Craftsmanship		Prentice Hall
Optional reading				
Eric Freeman, Elisabeth Robson, Bert Bates, Kathy Sierra	2004	Head First Design Patterns		O'Reilly Media
David Abrahams, Aleksey Gurtovoy		C++ Template Metaprogramming: Concepts, Tools, and Techniques from Boost and Beyond, Portable Documents		Addison-Wesley Professional