GDAŃSK UNIVERSITY OF TECHNOLOGY

Subject card

Subject name and code	Object-oriented programming languages II, PG_00064057		
Field of study	Technical Physics		
Date of commencement of studies	October 2026	Academic year of realisation of subject	2027/2028
Education level	first-cycle studies	Subject group	Optional subject group Subject group related to scientific research in the field of study
Mode of study	Full-time studies	Mode of delivery	at the university
Year of study	2	Language of instruction	Polish
Semester of study	4	ECTS credits	3.0
Learning profile	general academic profile	Assessment form	assessment
Conducting unit	Division of Theoretical Physics and Quantum Informaton -> Institute of Physics and Applied Computer Science -> Faculty of Applied Physics and Mathematics -> Faculties of Gdańsk University of Technology		
Name and surname of lecturer (lecturers)	Subject supervisor	dr hab. inż. arch. Jan Kozicki	
	Teachers		

Lesson types

Lesson type	Lecture	Tutorial	Laboratory	Project	Seminar	SUM
Number of study hours	0.0	0.0	45.0	0.0	0.0	45

E-learning hours included: 0.0

Learning activity and number of study hours

Learning activity	Participation in didactic classes included in study plan	Participation in consultation hours	Self-study	SUM
Number of study hours	45	4.0	26.0	75

Subject objectives

Student learns object-oriented programming in the selected programming language (C++ ISO/ANSI, C++14, C++17).

Learning outcomes

Course outcome	Subject outcome	Method of verification
[K6_U03] possesses programming skills in a selected language and the ability to use selected software packages.	The student is able to design and implement programs in an object-oriented programming language, applying object-oriented principles (e.g., encapsulation, inheritance, polymorphism) and using selected libraries and software development tools.	[SU1] Assessment of task fulfilment
[K6_W05] has knowledge of programming methodologies and techniques, as well as the use of selected IT tools in physics and engineering.	The student knows and understands object-oriented programming methods and techniques, including software design principles (e.g., encapsulation, inheritance, polymorphism, design patterns), as well as the possibilities of using selected software tools and libraries.	[SW1] Assessment of factual knowledge
[K6_K01] demonstrates readiness for continuous learning and updating knowledge in physics and related fields, critically evaluating it and recognising its importance in solving practical and theoretical problems.	The student is ready to independently update and expand their knowledge in object-oriented programming, critically evaluate available solutions and technologies, and apply them in programming tasks, recognizing the importance of continuous professional development.	[SK5] Assessment of ability to solve problems that arise in practice

<table>
<tr><td>Subject contents</td><td>Course content – laboratory

1. Basic elements of object-oriented design
2. Code reuse
3. Object-oriented analysis
4. Abstract data types
5. Classes and objects
6. Memory management
7. Inheritance mechanisms
8. Exception handling
9. Object-oriented design methodology</td></tr>
<tr><td>Prerequisites and co-requisites</td><td>Knowledge of operating systems Unix/Linux and MS Windows. Knowledge of the courses Procedural Programming Languages I (FIZ1C301) and II (FIZ1C307). Knowledge of the course Object-Oriented Programming Languages I (FIZ1C305).</td></tr>
<tr><td>Assessment methods and criteria</td><td>
<table>
<tr><th>Subject passing criteria</th><th>Passing threshold</th><th>Percentage of the final grade</th></tr>
<tr><td>Weekly short assignments based on lecture material from each week.</td><td>50.0%</td><td>45.0%</td></tr>
<tr><td>A written exam of the lecture and programming material</td><td>50.0%</td><td>45.0%</td></tr>
<tr><td>Very short tests of the practical skills of programming</td><td>50.0%</td><td>10.0%</td></tr>
</table>
</td></tr>
<tr><td>Recommended reading</td><td>
<table>
<tr><td>Basic literature</td><td>1) B. Stroustrup Programming Principles and Practice using C++, Addison Wesley</td></tr>
<tr><td>Supplementary literature</td><td>1. B. Meyer Object oriented software construction 2nd Ed.Prientice Hall PTR</td></tr>
<tr><td>eResources addresses</td><td></td></tr>
</table>
</td></tr>
<tr><td>Example issues/ example questions/ tasks being completed</td><td>1. Create a vector of Fibonacci numbers and print them using the functionfrom exercise 2. To create the vector, write a function, fibonacci(x,y,v,n),where integers x and y are ints, v is an empty vector, and n is thenumber of elements to put into v; v[0] will be x and v[1] will be y. A Fibo-nacci number is one that is part of a sequence where each element is thesum of the two previous ones. For example, starting with 1 and 2, we get1, 2, 3, 5, 8, 13, 21, ... Your fibonacci() function should make such asequence starting with its x and y arguments.

2. Define an Order class with (customer) name, address, data, andvector members. Purchase is a class with a (product) name,unit_price, and count members. Define a mechanism for reading andwriting Orders to and from a file. Define a mechanism for printing Or-ders. Create a file of at least ten Orders, read it into a vector, sortit by name (of customer), and write it back out to a file. Create anotherfile of at least ten Orders of which about a third are the same as in the firstfile, read it into a list, sort it by address (of customer), and writeit back out to a file. Merge the two files into a third using std::merge().

3. Write a binary search function for a vector (without using the standard one). You can choose any interface you like. Test it. How confidentare you that your binary search function is correct? Now write a binarysearch function for a list. Test it. How much do the two binarysearch functions resemble each other? How much do you think theywould have resembled each other if you had not known about the STL?

4. Modify the calculator from Chapter 7 minimally to let it take input froma file and produce output to a file (or use your operating systems facilitiesfor redirecting I/O). Then devise a reasonably comprehensive test for it.

5. What are the advantages and disadvantages of intrusive containers com-pared to C++ standard (non-intrusive) containers? Make lists of prosand cons.

6. Make a window (based on My_window) with a 4-by-4 checkerboard ofsquare buttons. When pressed, a button performs a simple action, such asprinting its coordinates in an output box, or turns a slightly different color(until another button is pressed).

7. explain keywords "this" and "constexpr"

8. what is the difference between static polymorphism and dynamic polymorphism. Explain with a code example using keywords "typename" and "virtual".</td></tr>
<tr><td>Practical activites within the subject</td><td>Not applicable</td></tr>
</table>