COURSE DESCRIPTION CARD - SYLLABUS

Course name
Algorithms and data structures 1 [S1Teleinf1>AiSD1]

Course

Field of study
Teleinformatics

Year/Semester
1/2

Area of study (specialization)
–

Profile of study
general academic

Level of study
first-cycle

Course offered in
Polish

Form of study
full-time

Requirements
compulsory

Number of hours

Lecture
30

Laboratory classes
30

Other
0

Tutorials
0

Projects/seminars
0

Number of credit points

5,00

Coordinators

dr inż. Filip Idzikowski
filip.idzikowski@put.poznan.pl

prof. dr hab. inż. Jerzy Tyszer
jerzy.tyszer@put.poznan.pl

Lecturers

Prerequisites

The student should have basic knowledge of discrete mathematics, combinatorics and probability theory. They should have the ability to perform calculations using mathematical apparatus in the field of mathematical analysis and probability, and to obtain information from the indicated sources.

Course objective

The course aims at introducing students to the area of algorithms and data structures. Furthermore, it presents methodologies and techniques of the object oriented programming using C++, providing a fairly complete introduction to the language.

Course-related learning outcomes

Knowledge:
The student has basic theoretical and practical knowledge of programming in C and C++, with an emphasis on designing well-formed programs, designing complex programs, and using library software. The student

also has knowledge of basic algorithms and data structures used in everyday programming practice.

Skills:
When designing software, the student can analyze a problem from an algorithmic perspective, applying criteria of computational complexity, scalability of the solutions used, and adequacy of the methods adopted. They can also critically analyze available library software for its application in a project and propose principles of collaboration within a collaborative programming configuration.

Social Competencies:
The student is aware of the possibilities and limitations of modern computer science while being open to possible applications in new areas of everyday life, economics, technology, and science.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Knowledge acquired during the lecture is assessed through a written exam consisting of several problem-solving exercises covering the lecture content and/or a multiple-choice test of approximately 15 questions. Skills acquired during laboratory exercises are assessed through two written tests covering tasks completed during the course. Furthermore, skills acquired during the course are continuously assessed through project exercises and oral presentations. Participation in the classes is also assessed.

Programme content

History of computational automation, software engineering, major categories of algorithms. C++ program structure, basic data types and instructions. Binary representation of integers, complement systems, bitwise operators. Computational complexity, N and NP problems, models of computation, the Turing machine. Functions, the role of the stack, argument passing, function overloading, function templates, lambda expressions. Recursion. Elementary sorting algorithms. Binary heaps and heap sort. Quick sort and merge sort. Information retrieval. Hash coding. Representation of floating-point numbers. Errors in numerical computation. Numerical equation solving. Finding the extremum of functions. Numerical integration. Gaussian elimination. Lexicographic ordering, permutation generation, transpositions, the Steinhaus-Johnson-Trotter algorithm, k-element subsets.

Course topics

Lecture: History of computational automation, software engineering, major categories of algorithms. C++ program structure, basic data types. Arithmetic and logical expressions, relations, casting, the statement block. Conditional statements, multivariate selection, loops, arrays. Binary representation of integers, the 2's complement system, bitwise operators, the Sieve of Eratosthenes. Computational complexity, N and NP problems, models of computation, the Turing machine. Functions, the role of the stack, argument passing, implicit arguments, function overloading, function templates, lambda expressions. Recursion, the Euclidean algorithm. Elementary sorting algorithms – insertion, selection, bubble sort, Shell's method. Binary heap and heap sort. Quick sort and merge sort. Binary and interpolation search. Hash coding. Choosing a hash function. Collisions and their avoidance. Representation of floating-point numbers. Errors in numerical computations. Numerical solution of equations. Bisection method, Newton-Raphson method. Finding the extremum of a function. Numerical integration. Gaussian elimination method. Lexicographic ordering, generating permutations, transpositions, Steinhaus-Johnson-Trotter algorithm, k-element subsets, random k-element subsets.

Labs: Familiarization with the programming system available in the lab. Simple file operations using an available text editor. Simple C++ programs: I/O operations, minimum/maximum search, matrix operations. Basic control statements. Functions – parameter passing, function templates, application examples. Recursion – simple recursive algorithms. Simple sorting algorithms (insertion, selection, bubble sort). Heap sort. Quicksort methods. Binary and interpolation search. Hash coding and collision resolution. Numerical methods: bisection, Newton-Raphson method. Generating simple combinatorial objects.

Teaching methods

Lectures: a multimedia presentation.

Laboratory classes: students solve various problems provided by a teacher, write programs, compile them,

debug a code, and evaluate programs on benchmark tests.

Bibliography

1. R. Sedgewick, Algorytmy w C++, Oficyna Wydawnicza READ ME, Łódź, 1999
2. N. Wirth, Algorytmy + struktury danych = programy, WNT, Warszawa, 1980.
3. T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Wprowadzenie do algorytmów, WNT, Warszawa, 2004
4. E.W. Dijkstra, Umiejętność programowania, WNT, Warszawa, 1985.
5. J. Grębosz, Symfonia C++, Oficyna Kallimach, Kraków 2008.
6. W. Lipski, Kombinatoryka dla programistów, WNT, Warszawa, 1982.

Breakdown of average student's workload

	Hours	ECTS
Total workload	120	5,00
Classes requiring direct contact with the teacher	64	3,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	56	2,00