COURSE DESCRIPTION CARD - SYLLABUS

Course name
Advanced Programming

Course

Field of study
Bioinformatics

Year/Semester
1/1

Area of study (specialization)
-

Profile of study
general academic

Level of study
Second-cycle studies

Course offered in
Polish

Form of study
full-time

Requirements
compulsory

Number of hours

Lecture	Laboratory classes	Other (e.g. online)
30	30	
Tutorials	Projects/seminars	

Number of credit points

5

Lecturers

Responsible for the course/lecturer:
dr inż. Marcin Radom
Faculty of Computing and Telecommunications

Responsible for the course/lecturer:
prof. dr inż. Marta Kasprzak
Faculty of Computing and Telecommunications

Prerequisites

Students commencing the Bioinformatics II degree program should have achieved the educational results of the first degree program, as defined in the Resolution of the Senate of PP - these results are presented on the department's website fct.put.poznan.pl.

In particular, students starting this course should have the knowledge and skills in the following subjects from the Bioinformatics I degree: Algorithms and Data Structures, Fundamentals of Programming, Object Oriented Programming.

Course objective

1. To provide students the knowledge of the .NET programming platform.

2. To introduce students to the construction principles of polynomial metaheuristic algorithms and exponential exact algorithms.

3. To teach students how to design and implement advanced algorithms for complex combinatorial problems derived from computational biology, using the .NET platform.

Course-related learning outcomes

Knowledge

As a result of the course, the student knows:

1. methods, techniques and tools used in the process of solving complex bioinformatics tasks, mainly of engineering nature,

2. principles of creating and testing complex software for bioinformatics problems,

3. principles of planning research in the development of new algorithmic solutions for computationally difficult bioinformatics problems.

Skills

As a result of the course, the student will be able to:

1. plan and perform advanced computational experiments and interpret their results,

2. apply advanced techniques and computer tools to solve biological problems and evaluate their usefulness,

3. discuss the results of their work in a scientific environment,

4. prepare a comprehensive written paper in Polish presenting the results of their own research,

5. apply a systematic approach to solving bioinformatics tasks, taking into account non-technical aspects.

Social competences

Successful completion of the course means that the student is ready to:

1. identify priorities to accomplish a task defined by self or others,

2. take responsibility for decisions made.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Formative assessment:

a) In the field of lectures, verification of the assumed educational effects is realized by answers to questions concerning the material discussed during the previous lectures.

b) In the field of laboratory, verification of the assumed educational effects is realized by:

- evaluation of skills related to the implementation of laboratory exercises,

- continuous assessment, at each class (oral answers), to reward the growth of skills in the use of known principles and methods,

- evaluation and "defense" of the laboratory exercises completed by the students,

Summing up assessment:

a) in the scope of lectures, verification of the assumed educational effects is realized by assessment of knowledge and skills related to the content provided during lectures in a form of one collective colloquium.

b) In the field of laboratory classes, the verification of the assumed educational effects is realized through the assessment of knowledge and skills related to the contents taught in lectures and laboratories, made on the basis of the final project presented by the student and the report (the grades given during the semester are taken into account).

Programme content

Lectures start with refreshing knowledge of C# language elements: the .NET programming environment, its architecture, basic .NET features and other C# related constructs. These include: program structure, data types and operations on them, program modularization - defining and calling functions, classes, object-oriented programming in C#: encapsulation, inheritance, polymorphism, interfaces, files and serialization. Collection classes and functions provided by them. Parametric collection classes (generics). Delegation: defining type, creating and using delegation object (delegation arrays, multiple delegations, anonymous delegations, delegation patterns). Lambda expressions. Events: defining an event associated with a delegation, registering functions in an event, reporting an event. Exception handling. Concurrency: creating concurrent threads and managing these threads. Concurrent execution of tasks, concurrent execution of calculations. Windows Forms library. Used namespaces, the initial form and its properties. Adding controls, defining their properties and defining event handling functions. Mouse and keyboard controls, menus, status bar, toolbars. Basic controls: buttons, text boxes, drop-down lists, etc. Creating graphs. Dialog boxes: standard and custom dialogs.

A series of lectures presenting ways to design algorithms begins with a reminder of the basic concepts of combinatorial optimization and an explanation of the terms heuristics and metaheuristics. Genetic algorithms: general principles, parameters, representation of an individual, adaptation function, initial population generation, selection, crossover, mutation, intensification and diversification strategy. Tabu search method: general principles, parameters, initial solution generation, definition of a move and a neighborhood, tabu list, aspiration criterion, memory types, intensification and diversification strategy. Heuristics: constructive vs. improving, greedy heuristics, beam search, problem decomposition, local search, hyperheuristics. Other metaheuristics: genealogy and classification, GRASP, evolutionary programming, genetic programming, scatter search, ant colony optimization, particle swarm optimization, hybrid heuristics. Exact algorithms: application constraints, branch & bound, branch & cut, dynamic programming.

Laboratory exercises are conducted in the form of 15 two-hour classes held in a computer laboratory. The first class is devoted to acquaint students with the rules of using the laboratory and the software. The program of laboratory classes includes the following issues: preparation of console applications in which basic elements of C# language are used, developing modularized programs using collection

classes, interfaces, delegations, developing programs using disk files, developing programs using Windows Forms library.

Within the laboratory classes a final project is realized. The project includes design, implementation and testing of one metaheuristic - genetic algorithm or tabu search method with polynomial time complexity - for one chosen bioinformatics problem from the given ones. The metaheuristics are to be specialized, i.e., configured to the properties of a particular problem in order to optimize the quality of the obtained solutions. In addition, the project includes the implementation of a procedure that generates non-trivial instances in a random manner, with parameters that allow the creation of instances of varying difficulty, especially large ones. This procedure will allow in-depth testing of the capabilities and limitations of metaheuristics, and the results of an extensive computational experiment are included in the report.

Teaching methods

1. Lecture: multimedia presentation, presentation illustrated with examples given if necessary.

2. Laboratory exercises: preparation of programs in .NET environment allowing to get to know particular technologies present in this environment; discussion of methods used for the completion of final project.

Bibliography

Basic

1. Pro C# 5.0 and the .NET 4.5 Framework, 6th edition, Andrew Troelsen, apress, 2012

2. Metaheuristics: From Design to Implementation, El-Ghazali Talbi, Wiley, Hoboken, 2009

3. Jak to rozwiązać czyli nowoczesna heurystyka, Zbigniew Michalewicz, David B. Fogel, WNT, Warszawa, 2006

Additional

1. Programowanie w C#, wydanie VI, Ian Griffiths, Matthew Adams, Jessy Liberty, O'Reilly 2010, Helion 2012

2. Algorytmy genetyczne i ich zastosowania, David E. Goldberg, WNT, Warszawa, 1995

3. Tabu Search, Fred Glover, Manuel Laguna, Kluwer Academic Publishers, Boston, 1997

4. Złożoność obliczeniowa problemów kombinatorycznych, Jacek Błażewicz, WNT, Warszawa, 1988

Breakdown of average student's workload

	Hours	ECTS
Total workload	125	5,0
Classes requiring direct contact with the teacher	60	2,5
Student's own work (preparation for laboratory classes, preparation for tests, project preparation) [1]	65	2,5

[1] delete or add other activities as appropriate