COURSE DESCRIPTION CARD - SYLLABUS

Course name
Computer architecture with low-level programming

Course

Field of study
Artificial Intelligence
Area of study (specialization)
-
Level of study
First-cycle studies
Form of study
full-time

Year/Semester
1/2
Profile of study
general academic
Course offered in
English
Requirements
compulsory

Number of hours

Lecture
15
Laboratory classes
15
Other (e.g. online)

Tutorials
Projects/seminars

Number of credit points

3

Lecturers

Responsible for the course/lecturer:
Tomasz Zok, Ph.D.
e-mail: tomasz.zok@cs.put.poznan.pl
tel: +48 61 665 3040
Faculty of Computing and Telecommunications
Piotrowo 2, 60-965 Poznan

Responsible for the course/lecturer:

Prerequisites

The student should have the ability to obtain information from indicated sources and show willingness to work in a team.

Course objective

To provide knowledge about low-level aspects of programming in C. Developing students' awareness of challenges and potential difficulties while designing low-level applications. Familiarize students with the x86 CPU architecture and extensions.

Course-related learning outcomes

Knowledge

1. has a well structured knowledge of programming in C.

2. is familiar with the most common errors and problems associated with designing low-level applications.

3. is familiar with x86 computer system architecture and extensions.

Skills

1. is able to design and implement a C-language application that solves problems such as processing text or binary data.

2. is able to use a debugger and memory leakage analysis programs to solve the most common problems related to low-level application development.

3. is able to create an x86 assembly application including optimizing the use of SIMD operations.

Social competences

1. is aware that knowledge of computer system architecture and the ability to create low-level applications translates into a fuller understanding of any type of IT solutions.

2. understands that low-level solutions are crucial from the point of view of IT systems security.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Verification of the knowledge acquired in the course of the lecture is done by means of a written assessment containing open or multiple choice questions. The pass threshold is 50%.

Skills acquired during the laboratory classes are verified by evaluating several projects or practical tasks. Students can work in pairs. Each project is evaluated separately. In order to pass the laboratory classes it is required to receive at least a 3.0 grade from each project.

Programme content

Lecture:

1. C, Basics, Types, Literals, Expressions, Statements

2. Functions, Arrays, Pointers

3. Structures, Unions and Bit-Fields

4. Dynamic Memory Management, Input and Output

5. Multithreading, Floating-point Numbers

6. Computer Architecture

7. x86 Assembly

8. Test

Laboratory classes:

1. Practical guide to debugging

2-3. Project 1: Text File Parsing

4-5. Project 2: Binary File Parsing

6-7. Project 3: x86 Assembly

8. Late Project Reporting

Teaching methods

Lecture: multimedia presentation

Laboratory exercises: multimedia presentation, developing examples at the board, working in pairs

Bibliography

Basic

Peter Prinz, Tony Crawford „C in a nutshell”

David Patterson, John Hennessy „Computer organization and design”

Additional

Gynvael Coldwind „Zrozumieć programowanie”

Breakdown of average student's workload

	Hours	ECTS
Total workload	75	3
Classes requiring direct contact with the teacher	30	1,2
Student's own work (literature studies, preparation for laboratory classes/tutorials, preparation for tests/exam, project preparation) [1]	45	1,8

[1] delete or add other activities as appropriate