Low-level Language Programming

Course description

Basic information

Field of study : Analytical Computer Science

Path : -

Organizational unit : Faculty of Mathematics and Computer Science

Level of education : first-cycle studies

Form of studies : full-time studies

Study profile : general academic

Mandatory status : mandatory

Education cycle : 2022/23

Course code : UJ.WMIIANS.180.03299.22

Languages of instruction : Polish

Disciplines : Computer Science

ISCED classification : 0613 Software and applications development and analysis

USOS code : WMI.TCS.PN.OL

Course coordinator

Jakub Kozik

Course instructor

Jakub Kozik, Jan Derbisz

Period Semester 4	Form of verification of learning outcomes exam Form of teaching and hours lecture: 30 laboratory classes: 30	Number of ECTS credits 6.0

Educational goals for the course

C1	When we exhaust algorithmic ways to speed up a computer program, we are left with low-level code optimization. Compilers, virtual machines, and code generators become more powerful every year, but to achieve the highest efficiency, we often need to optimize code manually. In this course, you will learn how to manage memory well, why it's worth using cursor structures, and how to live in harmony with cache memory. We will show you how not to ruin the performance of a multithreaded program with poor synchronization. We will teach you how to squeeze every last bit of performance from modern processors using their vector capabilities and many other useful techniques that will expand your programming toolkit.

Learning outcomes for the course

Code	**Effects in the area of**	**Major learning outcomes**	**Verification methods**
Knowledge – The student knows and understands:			
W1	fundamentals of modern processor architecture.	IAN_K1_W13	written exam, graded credit
Skills – The student can:			
U1	program in assembly language using advanced processor functionalities (including SIMD and atomic instructions).	IAN_K1_U03, IAN_K1_U09	written exam, graded credit, programming tasks
U2	use elements of low-level optimization in high-level language programming (C/C++).	IAN_K1_U03, IAN_K1_U05, IAN_K1_U09	written exam, graded credit, programming tasks

ECTS credits balance

Student activity form	Average number of hours* dedicated to completed activity types
lecture	30
laboratory classes	30
independent solving of computer tasks	75
exam preparation	40

Total student workload	Number of hours 175	ECTS credits 6.0

* hour (lesson) means 45 minutes

Course content

No.	Course content	Learning outcomes for the course
1.	Basics of x86_64 architecture.	W1
2.	Low-level interfaces of Linux system and C and C++ languages.	W1, U1
3.	Elements of processor microarchitecture (including pipeline processing and cache operation).	W1, U2
4.	SIMD (Single Instruction Multiple Data) instructions.	W1, U1, U2
5.	Binary code and programs modifying code (Self Modifying Code).	U1
6.	Low-level aspects of multithreaded programming.	W1, U2

Extended information

Teaching methods:

multimedia lecture, discussion, problem solving, laboratory classes

Type of classes	Forms of credit	Course credit requirements
lecture	written exam	The condition for passing the course is obtaining a positive grade from classes and passing the exam with more than 50% of points.
laboratory classes	graded credit, programming tasks	The condition for passing the classes is obtaining more than 50% of points for completing programming tasks. Activity during classes can improve the grade but does not affect the fact of passing.

Prerequisites and additional requirements

- Ability to program in C and C++. - Knowledge of operating systems basics.

Literature

Required

1. Technical documentation for x86_64 architecture for AMD or Intel processors, available on the manufacturer's website.

Additional

1. Randall Hyde, The Art of Assembly Language Programming (available on the author's website)
2. Agner Fog, Optimization manuals (available on the author's website)