COURSE DESCRIPTION CARD - SYLLABUS

Course name

System and concurrent programming

Course

Field of study	Year/Semester
Computing	2/3
Area of study (specialization)	Profile of study
-	general academic
Level of study	Course offered in
First-cycle studies	Polish
Form of study	Requirements
full-time	compulsory

Number of hours

Lecture	Laboratory classes	Other (e.g. online)
30	30	0
Tutorials	Projects/seminars	
0	0	

Number of credit points

5

Lecturers

Responsible for the course/lecturer:	Responsible for the course/lecturer:
Prof. dr hab. inż. J. Brzeziński	dr hab. inż. Anna Kobusińska
email: Jerzy.Brzezinski@cs.put.poznan.pl	email: Anna.Kobusinska@cs.put.poznan.pl
tel. tel. (0-61) 665-2903, fax: (0-61) 877 1525,	tel. tel. (0-61) 665-2964, fax: (0-61) 877 1525,
Instytut Informatyki	Instytut Informatyki
ul. Piotrowo 2, 60-965 Poznań	ul. Piotrowo 2, 60-965 Poznań

Prerequisites

Students starting this course should have basic knowledge of the functioning of operating systems presented in the Operating Systems course. They should also have the following skills: programming, defining low-level data structures and solving basic problems of low-level coding of algorithms acquired in the course of Low-level programming. Students should also have the ability to obtain information from the indicated sources, as well as understand the need to expand their competences and be ready to cooperate as part of the team.

Course objective

The objective for this course is to give the students knowledge in the field of concurrent programming, process management, synchronization mechanisms and deadlock prevention. Moreover, during the course the synchronization mechanisms that are used to solve classical synchronization problems are disscussed.

Course-related learning outcomes

Knowledge

1. Students posesses well-grounded knowledge on key issues in the field of system and concurrent programming, and the detailed knowledge in the field of operating systems

2. Students have basic knowledge of the life cycle of operating systems, in particular about the principles of process management, synchronization mechanisms and deadlock detection

3. knows the basic techniques, methods and tools used in the process of solving IT engineering tasks in the field of system and concurrent programming

Skills

1. Students are able to formulate and solve IT tasks, use appropriately selected methods of system and concurrent programming, including analytical methods

2. Students are able to assess the computational complexity of concurrent algorithms

3. Students can - in accordance with the given specification - design (formulate the functional specification and non-functional requirements for selected quality characteristics) and implement a broadly understood IT systems, selecting a programming language appropriate for a given programming task and using appropriate methods, techniques and tools of concurrent programming

4. Students have the ability to formulate concurrent algorithms and implement them

Social competences

1. Students understand the importance of using the latest knowledge from the field of computer science in solving research and practice problems

2. Students are aware of the importance of knowledge in the field of system and concurrent programming in solving engineering problems and know examples of malfunctioning IT systems that have led to serious financial and social losses

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

The knowledge acquired during lectures is verified during a problem-based written exam that consists of open and/or test questions. The maximum number of points per open question is 10, and per test question is 1. To pass the exam students must obtain at least 50% of the total points.

The skills acquired during the exercises are verified in the following way:

- assessment of the students' preparation for classes ("entrance" test),

- continuous assessment during each class (oral answers),

- assessment of knowledge and skills obtained during the project that is implemented as a homework

It is possible to get additional points for activity during classes, especially during discussing additional aspects of the considered problems. Passing score: 50% of total points.

Programme content

Lectures cover the following topics:

1.Elements of concurrent programming (process flow graphs, and notations "and", "fork-join-quit", "parbegin-parend")

2) Petri nets graphs and the application of Petri nets in modeling of concurrent processes

3) The problem of mutual exclusion and exemplary software methods of solving it, including, among others, the algorithms: Dekker, Dijkstra, Peterson for two and n processes, Lamport

4) The synchronization mechanisms: hardware (test-and-set instructions, active wait, blocking the interrupt system), system (binary and general semaphores, lock and unlock operations, enq and deq operations, wait and post operations, block and wakeup operations) , event counters), software (critical regions, conditional critical regions, monitors, software implementations of synchronization mechanisms) and communication (synchronous and asynchronous send and receive messages exchange).

5) The application of selected synchronization mechanisms to solve classical problems of synchronization (mutual exclusion, producer-consumer problem, reader-writer problem, problem of five philosophers)

6) Process management: the concept of a process, process state graph, the problem of scheduling tasks in probabilistic and deterministic terms (ranking evaluation criteria), scheduling algorithms.

7) The definition of a deadlock, necessary and sufficient conditions for deadlocks, deadlock prevention (prevention, avoidance, detection and elimination approach).

During the laboratory classes, students implement the mechanisms offered by the UNIX kernel, and use the information obtained during the lectures to implement algorithms and synchronization mechanisms. The following topics are discussed in the laboratories:

1) File operations

2) Process handling: creating and deleting processes, running programs, redirecting standard streams: input, output and diagnostic output

3) Creation and handling of named and unnamed pipes, examples of errors in synchronization of processes using pipes

4) IPC mechanisms: access to shared memory, support for semaphores and message queues. Use of known mechanisms for process synchronization; implementation of algorithms learned during the lectures with the use of selected IPC mechanisms

5) Thread handling and management

Teaching methods

1. Lectures: multimedia presentation, illustrated with examples given on the blackboard.

2. Laboratory classes: a multimedia presentation illustrated with examples and practical excersises, project.

Bibliography

Basic

1. Operating Systems: Design and Implem., Tanenbaum A., Prentice-Hall Intern. Ed., 2008

2. Podstawy systemów operacyjnych, Silberschatz A., Galvin P.B., WNT, 2006

3. Operating System Concepts, 8th, Update Edition, Abraham Silberschatz, Peter B. Galvin, Greg Gagne, Wiley&Sons, 2011

4. Program. w systemie Unix dla zaawansowanych, Marc J. Rochkind, WNT, 2008

5. System operacyjny LINUX, Cezary Sobaniec, Nakom, 2002

6. Unix i Linux. Przewodnik administratora systemów. Wydanie IV, E. Nemeth, i inni, WNT, 2011

Additional

1. Operating Systems - A Modern Perspective, 3rd Edition , Nutt, G.J, Addison-Wesley Pub, 2003

2. Operating Systems, 3/E, Deitel I inni, Prentice Hall Intern, 2004

3. The Linux Programming Interface, Michael Kerrisk, No Starch Press, 2010

4. Advanced Programming in the Unix Environment (3rd Edition), R.Stevens, S.Rago, O'Reilly, 2013

5. Linux System Programming: Talking Directly to the Kernel and C Library, R. Love, O'Reilly, 2007

6. Linux Kernel Development, R. Love, Addison-Wesley, 2010

7. Operating Systems: Internals and Design Principles (8th Edition), Stallings W., Prentice Hall Intern, 2018

Breakdown of average student's workload

	Hours	ECTS
Total workload	120	5,0
Classes requiring direct contact with the teacher	64	3,0
Student's own work (literature studies, preparation for laboratory classes/tutorials, preparation for tests/exam, project preparation) [1]	56	2,0

[1] delete or add other activities as appropriate