COURSE DESCRIPTION CARD - SYLLABUS

Course name
System and concurrent programming [N1Inf1>PSW]

Course

Field of study
Computing

Year/Semester
2/3

Area of study (specialization)
–

Profile of study
general academic

Level of study
first-cycle

Course offered in
Polish

Form of study
part-time

Requirements
compulsory

Number of hours

Lecture
20

Laboratory classes
20

Other
0

Tutorials
0

Projects/seminars
0

Number of credit points

5,00

Coordinators

dr inż. Dariusz Wawrzyniak
dariusz.wawrzyniak@put.poznan.pl

Lecturers

Prerequisites

A student starting this module should have basic knowledge of computer functioning and imperative programming. He should have the ability to solve basic problems in the field of implementation and assessment of the cost of running simple algorithms and the ability to obtain information from indicated sources. He should also understand the need to expand his competences and be willing to cooperate within a team. Moreover, in terms of social competences, the student must demonstrate attitudes such as honesty, responsibility, perseverance, cognitive curiosity, creativity, personal culture, and respect for other people.

Course objective

The purpose of the item is:
- providing students with basic theoretical knowledge related to operating system kernel services and concurrent processing,
- familiarizing students with practical aspects of implementing concurrent processing,
- developing students' skills in solving problems related to concurrent processing in computer systems.

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:

Formative assessment:
1.
2. In the scope of lectures, verification of the assumed learning outcomes is carried out by:
   - answers to questions about the material discussed during lectures.
3.
4. In the field of laboratories, verification of the assumed learning outcomes is carried out by:
   - assessment of the student's preparation for individual laboratory classes,
   - assessment of skills related to the implementation of laboratory exercises.

Summary rating:
1.
2. In the scope of lectures, verification of the assumed learning outcomes is carried out by:
   - assessment of knowledge and skills demonstrated in a test-type assessment, consisting of several open or closed questions/tasks, with the possibility of obtaining a total of 100 points with a threshold of 50 points for a positive grade,
   - discussion of the test results.
3. In the field of laboratories, verification of the assumed learning outcomes is carried out by:
   - assessment of knowledge and skills related to the content provided in the laboratories through the final colloquium,
   - list of grades awarded during the semester in the form of an average.

Activity during classes is rewarded with additional points, in particular for:

- discussion of additional aspects of the issue,
- effectiveness of applying acquired knowledge when solving a given problem,
- comments leading to the improvement of teaching materials or the teaching process.

Programme content

The module program covers the following topics:

1. Concurrent programming abstraction: atomic operations and their interleaving.
2. Processes and threads.
3. .
4. The concepts of security and lifeness.
5. The problem of mutual exclusion.
6. Complex atomic operations.
7. Synchronization i interprocess communication mechanisms supported by the operating system.
8. Classic problems of synchronization.
9. Language support for synchronization mechanisms: conditional critical regions, monitors.
10. Deadlock problem: deadlock definition, necessary and sufficient conditions for deadlock, deadlock prevention (prevention, avoidance, detection and resolution).

Course topics

The lecture program covers the following topics:

1. Concurrent programming abstraction: introducing the concept of atomic operations and their interleaving.
2. The problem of mutual exclusion and its solution based on atomic operations of reading and writing shared variables (e.g. algorithms: Peterson, Lamport), the concepts of security and lifetime.
3. Complex atomic operations, e.g.: test-and-set, exchange.
4. Synchronization mechanisms supported by the operating system: binary and counting semaphores, POSIX standard mechanisms (locks and conditional variables).
5. Classic problems of synchronization: producer-consumer, readers and writers, five philosophers, sleeping hairdressers.
6. Language support for synchronization mechanisms: conditional critical regions, monitors.
7. Deadlock problem: deadlock definition, necessary and sufficient conditions for deadlock, deadlock prevention (prevention, avoidance, detection and resolution).

The laboratory curriculum covers the following issues at the level of the system services interface in C:

1. File access.

4. Process management.
5.
6. Signal handling.
7.
8. Inter-process communication and synchronization via links.
9.
10. Inter-process communication and synchronization via IPC mechanisms.
11.

Teaching methods

Lecture: multimedia presentation, presentation illustrated with examples given on the board, solving tasks. Laboratory exercises: classes in a computer laboratory involving the implementation of programs (including concurrent ones) based on the functions of the operating system kernel, discussion.

Bibliography

Basic:
1.
2. Mordechai Ben-Ari, Podstawy programowania współbieżnego i rozproszonego, WNT, W-wa, 2009.
3.
4. A. Silberschatz, J.L. Peterson, G. Gagne, Podstawy systemów operacyjnych, WNT, W-wa, 2006.
5.

Additional:
1.
2. Z. Weiss, T. Gruźlewski, Programowanie współbieżne i rozproszone w przykładach i zadaniach, WNT, W-wa, 1993.
3.

Breakdown of average student's workload

	Hours	ECTS
Total workload	125	5,00
Classes requiring direct contact with the teacher	42	2,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	83	3,00