



# System Software

## Syllabus

### Requisites of the Course

Cycle of Higher Education	First cycle of higher education (Bachelor's degree)
Field of Study	12 Information Technologies
Speciality	123 Computer Engineering
Education Program	Computer Systems and Networks
Type of Course	Normative
Mode of Studies	Full-time
Year of studies, semester	3 year, 5 semester
ECTS workload	5.5 credits (ECTS). Time allotment – 165 hours, including 36 hours of lectures, 18 hours of laboratory classes and 111 hours of self-study.
Testing and assessment	Exam
Course Schedule	1.5 classes per week by the timetable <a href="http://roz.kpi.ua/">http://roz.kpi.ua/</a>
Language of Instruction	English
Course Instructors	Lecturer: senior lecturer, Andrey Simonenko, <a href="mailto:comsys.spz@gmail.com">comsys.spz@gmail.com</a> Teacher of laboratory works: senior lecturer, Andrey Simonenko, <a href="mailto:comsys.spz@gmail.com">comsys.spz@gmail.com</a>
Access to the course	<a href="https://classroom.google.com/c/NjMzNTQ4NTY2NTQx?cjc=gjeeewf">https://classroom.google.com/c/NjMzNTQ4NTY2NTQx?cjc=gjeeewf</a>

### Outline of the Course

#### 1. Course description, goals, objectives, and learning outcomes

The discipline “System Software” is aimed at studying the principles of functioning, architecture and implementation of kernels of general purpose operating systems. Implementations of Unix-like operating system kernels and implementations of other kernel types are taken as a basis, that is, the discipline is not focused on any one specific operating system. Thorough information on tasks and methods of solving them in the operating system kernel at a low level of implementation is provided. The discipline consists of the following parts: kernel and processes, support for multithreaded programs, file systems, memory management.

The discipline provides the following competencies and program learning outcomes of the educational and professional program “Computer Engineering”:

3K1 Ability to abstract thinking, analysis and synthesis.

3K2 Ability to learn and master modern knowledge.

ФК12 The ability to identify, classify and describe the functioning of software and technical means, computer and cyber-physical systems, networks and their components by using analytical and modeling methods.

ПРН8 Be able to think systematically and apply creative abilities to the formation of new ideas.

The discipline should prepare students to understand parts of the source code of existing general-purpose operating system kernels that implement process and thread management, system calls, file systems, and memory management.

According to the results of studying the discipline, the student should have an understanding of functioning of the general-purpose operating system kernel, which will allow working with parts of the

source code of existing general-purpose operating system kernels, developing more efficient system and application programs.

## **2. Prerequisites and post-requisites of the course (the place of the course in the scheme of studies in accordance with curriculum)**

Required disciplines: Programming, Data Structures and Algorithms, Computer Architecture, System Programming.

Understanding Assembler, understanding C or C++ to read examples in lectures.

## **3. Content of the course**

Part 1. Kernel and processes

Topic 1.1. Definition of the kernel

Topic 1.2. Execution modes and contexts

Topic 1.3. Switching to kernel mode

Topic 1.4. Hardware interrupts handling

Topic 1.5. System calls

Topic 1.6. Involuntary context switching

Topic 1.7. Voluntary context switching

Topic 1.8. Signals

Topic 1.9. Monolithic kernel and microkernel

Topic 1.10. Process control

Part 2. Support for multithreaded programs

Topic 2.1. Multithreaded program and synchronization

Topic 2.2. N:1 mode

Topic 2.3. 1:1 mode

Topic 2.4. N:M mode

Part 3. File systems

Topic 3.1. Definition of a file system

Topic 3.2. Virtual file system

Topic 3.3. Stackable file systems

Topic 3.4. File system for flash memory

Topic 3.5. Requirements for local file systems

Part 4. Memory management

Topic 4.1. Static relocation, memory segmentation

Topic 4.2. Virtual memory

Topic 4.3. Page replacement algorithms

Topic 4.4. Shared memory

## **4. Coursebooks and teaching resources**

1. Andrew S. Tanenbaum, Herbert Bos. *Modern Operating Systems*, 4th edition - Pearson, 2014. - 1136 p.
2. Daniel P. Bovet, Marco Cesati. *Understanding the Linux Kernel*, 3rd edition - O'Reilly Media, 2005. - 944 p.
3. Richard McDougall, Jim Mauro. *Solaris Internals: Solaris 10 and OpenSolaris Kernel Architecture*, 2nd edition - Prentice Hall, 2006. - 1020 p.
4. Chris Cooper, Chris Moore. *HP-UX 11i Internals* - Prentice Hall, 2004. - 432 p.
5. Amit Singh. *Mac OS X Internals: A Systems Approach* - Addison-Wesley Professional, 2006. - 1641 p.

## Educational content

### 5. Methodology

Names of parts and topics	Number of hours			
	In total	Including		
		Lectures	Laboratory classes	Self-study
Part 1. Kernel and processes				
Topic 1.1. Definition of the kernel	5	1		4
Topic 1.2. Execution modes and contexts	4	1		3
Topic 1.3. Switching to kernel mode	4	1		3
Topic 1.4. Hardware interrupts handling	4	1		3
Topic 1.5. System calls	4	1		3
Topic 1.6. Involuntary context switching	4	1		3
Topic 1.7. Voluntary context switching	4	1		3
Topic 1.8. Signals	4	1		3
Topic 1.9. Monolithic kernel and microkernel	4	1		3
Topic 1.10. Process control	4	1		3
Part 2. Support for multithreaded programs				
Topic 2.1. Multithreaded program and synchronization	17	2		15
Topic 2.2. N:1 mode	5	2		3
Topic 2.3. 1:1 mode	5	2		3
Topic 2.4. N:M mode	5	2		3
Part 3. File systems	24		12	12
Topic 3.1. Definition of a file system	6	2		4
Topic 3.2. Virtual file system	5	2		3
Topic 3.3. Stackable file systems	5	2		3
Topic 3.4. File system for flash memory	5	2		3
Topic 3.5. Requirements for local file systems	5	2		3
Part 4. Memory management	10		6	4
Topic 4.1. Static relocation, memory segmentation	7	2		5
Topic 4.2. Virtual memory	10	2		8
Topic 4.3. Page replacement algorithms	8	2		6
Topic 4.4. Shared memory	7	2		5
Exam	6			
In total	165	36	18	111

#### Lectures

Lecture	The topic of the lecture and a list of main questions (a list of didactic tools, references to the literature and tasks for the self-study)
1	<p><b>Definition of the kernel. Execution modes and contexts.</b></p> <p>Program execution environment. Components of OS. Two parts of the kernel. Interaction of a user program with the kernel. Tasks of the kernel. The kernel is also a program. Loading of the kernel into the memory. OS classification by purpose. Safe execution of user programs. CPU privilege levels. Execution modes. Execution contexts. Process context.</p>

	Self-study: Read relevant topics in the literature.
2	<p><b>Switching to kernel mode. Hardware interrupt handling.</b>  Conditions for switching to kernel mode. Hardware interrupts. Software interrupts. Exceptions. Memory pyramid. Programmable interrupt controller. Hardware interrupt handling, the first variant. Hardware interrupt handling, the second variant. Hardware interrupt handling, the third variant. Polling.  Self-study: Read relevant topics in the literature.</p>
3	<p><b>System calls. Involuntary context switching.</b>  Definition of a system call. Invoking a system call. Virtual system calls. Binary compatibility mode. Simultaneous execution of processes. Implementation of involuntary context switching. Tickless kernel.  Self-study: Read relevant topics in the literature.</p>
4	<p><b>Voluntary context switching. Signals.</b>  Resource concept. Implementation of voluntary context switching. Resuming execution of a blocked process. Definition of a signal. Sending a signal. Delivering a signal.  Self-study: Read relevant topics in the literature.</p>
5	<p><b>Monolithic kernel and microkernel. Process control.</b>  Two paradigms of kernel architecture. Monolithic kernel. Microkernel. Process state diagram. Creation of a new process. Termination of a process.  Self-study: Read relevant topics in the literature.</p>
6	<p><b>Multithreaded program and synchronization.</b>  Definition of a multithreaded program. Reasons for developing a multithreaded program. Speed-up of a multithreaded program. Synchronization.  Self-study: Read relevant topics in the literature.</p>
7	<p><b>N:1 mode.</b>  Definition of N:1 mode. Implementation of voluntary context switching. Implementation of involuntary context switching. Implementation of synchronization mechanisms. Advantages and disadvantages.  Self-study: Read relevant topics in the literature.</p>
8	<p><b>1:1 mode.</b>  Definition of 1:1 mode. "Multithreaded" kernel. CPU affinity. Priority inversion. Task, process, thread. Advantages and disadvantages.  Self-study: Read relevant topics in the literature.</p>
9	<p><b>N:M mode.</b>  Definition of N:M mode. Invoking the kernel scheduler. Invoking the user scheduler. Advantages and disadvantages.  Self-study: Read relevant topics in the literature.</p>
10	<p><b>Definition of a file system.</b>  Definition. File descriptor. File types. File name (except for directory). Directory name. Pathname. FS tree. Device file.  Self-study: Read relevant topics in the literature.</p>
11	<p><b>Virtual file system.</b>  Definition. Variant of implementation. Five types of file descriptors. Buffer cache. File name cache. Mounting.  Self-study: Read relevant topics in the literature.</p>
12	<p><b>Stackable file systems.</b>  Definition of a stackable FS. Variant of implementation. Examples of stackable FSes. Definition</p>

	of a synthetic FS. Self-study: Read relevant topics in the literature.
13	<b>File system for flash memory.</b> Properties of NAND flash memory. Journaling FS. Optimization variants. Self-study: Read relevant topics in the literature.
14	<b>Requirements for local file systems.</b> Extent. Checksums. Data encryption. Data compression. Arrays. Mirrors. Deduplication. Snapshots. Journaling. Self-study: Read relevant topics in the literature.
15	<b>Static relocation, memory segmentation.</b> Static relocation. Memory segmentation. Swapping. Overlay. Self-study: Read relevant topics in the literature.
16	<b>Virtual memory.</b> Definition of virtual memory. Physical and virtual pages. Page table. Inverted page table. Hashed page table. Process address map. Translation lookaside buffer. Self-study: Read relevant topics in the literature.
17	<b>Page replacement algorithms.</b> Working set. Page replacement algorithms. Paging, demand paging, predefault. Thrashing. Local and global policies. Page coloring. Superpages. Self-study: Read relevant topics in the literature.
18	<b>Shared memory.</b> Local shared memory. Distributed shared memory. Idea of copy-on-write. Implementation of copy-on-write. Zero copy. Memory guarantee strategies. Self-study: Read relevant topics in the literature.

### Laboratory tasks

Choose three laboratory tasks from the five ones proposes, each laboratory tasks needs the same amount of time to complete.

No	Laboratory task
1	General purpose memory allocator using tags
2	General purpose memory allocator using slab allocation
3	Page replacement algorithms
4	File system, part 1
5	File system, part 2

### 6. Self-study

In the process of understanding topics from lectures and performing laboratory works students must consolidate the knowledge gained during lectures and practical work, self-study certain topics using information from Internet, deepen their knowledge for further study.

Self-study is the following:

1. Studying and understanding topics from previous lectures.
2. Performing tasks given for self-study.
3. Performing laboratory works.
4. Writing reports for laboratory works.

### 7. Course policy

Course policy completely corresponds to rules and regulations published by KPI. Each student performs each laboratory work personally. If a laboratory work was not performed by the student personally (even if something unprincipled is changed in the solution), then it is not credited.

### 8. Monitoring and grading policy

According to regulations published by KPI maximum number of 100 points is divided between laboratory works and exam. It is possible to get a total of 45 points for completing three laboratory works and 55 points for answering questions on the exam. The maximum score for each laboratory work is 15. It is necessary to choose three laboratory works from the five proposed ones.

In order to receive credit for a laboratory work, it is necessary to score 60% of the maximum number of points for it. To be admitted to the exam, it is necessary to pass all laboratory works. To receive the first attestation, one laboratory work must be credited. To receive the second attestation, two laboratory works must be credited. The first laboratory work is graded with 1 penalty point after the completion of the first attestation, graded with 2 penalty points after the completion of the second attestation. The second laboratory work is graded with 1 penalty point after the completion of the second attestation. Any laboratory work is graded with 2 penalty points after the completion of the main grading session. The number of attempts to pass any laboratory work is limited to two attempts, penalty points are not charged for the first attempt that is not credited. Reports on completed laboratory work are submitted and checked according to the schedule of the group's classes. If the completed laboratory work contains errors or inconsistencies with the conditions of the task, then the laboratory work is not credited or is credited with lower points (the decision on the points is made by the teacher and reported to the author of the work).

The exam consists of four questions, and the maximum number of possible points for the exam is evenly distributed among these four questions. The exam is verbal.

The final performance score or the results of the Pass/Fail are adopted by KPI grading system as follows:

Score	Grade
100-95	Excellent
94-85	Very good
84-75	Good
74-65	Satisfactory
64-60	Sufficient
Below 60	Fail
Course requirements are not met	Not Graded

### Syllabus of the course

Is designed by teacher senior lecturer, **Andrey Simonenko**

Adopted by Department of Computing Technics (protocol № 13 10.05.2023)

Approved by the Faculty Board of Methodology (protocol № 11 29.06.2023)