Federal State Autonomous Educational Institution of Higher Education "Moscow Institute of Physics and Technology (National Research University)"

APPROVED
Head of the Phystech School of Applied Mathematics and Informatics
A.M. Raygorodskiy

Work program of the course (training module)

course:	Operating Systems II/Операционные системы II
major:	Information Science and Computer Engineering
specialization:	Computer Science/Информатика
	Phystech School of Applied Mathematics and Informatics
	Chair of Algorithms and Programming Technologies
term:	2
qualification:	Bachelor

Semester, form of interim assessment: 4 (spring) - Grading test

Academic hours: 60 AH in total, including:
    lectures: 30 AH.
    seminars: 0 AH.
    laboratory practical: 30 AH.

Independent work: 120 AH.

In total: 180 AH, credits in total: 4

Author of the program: V.V. Yakovlev, candidate of physics and mathematical sciences

The program was discussed at the Chair of Algorithms and Programming Technologies 21.05.2020

Annotation

The course studies low-level programming for UNIX-like operating systems. The primary programming language is plain C, but not C++, to study programming techniques closely related to operating system's components.

The first part of course related to system calls conceptions and low-level programming from the assembly code point of view. Second part of course covers various mechanisms of inter-process communications: signals, shared memory etc.

1. Study objective

Purpose of the course

> The purpose of the course is the development of fundamental knowledge in the field of operating systems by students.

Tasks of the course

> - Formation of a general idea of the basic concepts in the field of operating systems and methods of their design;
> - the formation of basic knowledge about the main objects of operating systems and how to work with them;
> - teaching students examples of practical implementation of operating systems.

2. List of the planned results of the course (training module), correlated with the planned results of the mastering the educational program

Mastering the discipline is aimed at the formation of the following competencies:

Code and the name of the competence	Competency indicators
Gen.Pro.C-1 Apply fundamental knowledge acquired in the physical and mathematical fields and/or natural sciences and use it in professional settings	Gen.Pro.C-1.1 Analyze the task in hand, outline the ways to complete it

3. List of the planned results of the course (training module)

As a result of studying the course the student should:

know:

> - Main types of objects of operating systems;
> - methods for implementing objects of operating systems;
> - structure of operating systems using Linux as an example;
> - means of parallelization of calculations of the POSIX standard;
> - methods of real-time implementation in the OS;
> - methods of organizing OS design work;
> - testing methods for large software systems.

be able to:

> - Use the basic mechanisms of OS such as Linux;
> - synchronize parallel computing.

master:

> - Basic mechanisms for working with large software systems.

4. Content of the course (training module), structured by topics (sections), indicating the number of allocated academic hours and types of training sessions

4.1. The sections of the course (training module) and the complexity of the types of training sessions

№	Topic (section) of the course	Types of training sessions, including independent work			
		Lectures	Seminars	Laboratory	Independent

		Lectures	Seminars	practical	work
1	Process intercommunication using pipe channels	6		6	24
2	FIFO channels and file locks	6		6	24
3	Signals and theirs handling	6		6	24
4	Process intercommunication using mapped memory regions	6		6	24
5	POSIX shared memory and POSIX semaphores	6		6	24
AH in total		30		30	120
Exam preparation		0 AH.			
Total complexity		180 AH., credits in total 4			

4.2.   Content of the course (training module), structured by topics (sections)

Semester: 4 (Spring)

1. Process intercommunication using pipe channels

Understanding RPC and passing parameters through the universal XDR view. Libdoor library.

2. FIFO channels and file locks

Using the make utility to build large software systems. GNU make recommendations.

3. Signals and theirs handling

Svn utility.

4. Process intercommunication using mapped memory regions

Test requirements. Testing Management Systems.

5. POSIX shared memory and POSIX semaphores

Software packages as software delivery units. Organization of distributions on the example of Debian.

5. Description of the material and technical facilities that are necessary for the implementation of the educational process of the course (training module)

Necessary equipment for lectures: computer and multimedia equipment (projector). Required software: operating system with an Internet browser. Self-service: access to science libraries.

6. List of the main and additional literature, that is necessary for the course (training module) mastering

Main literature
1. Linux [Текст] / А. А. Стахнов .— СПб : БХВ-Петербург, 2003 .— 912 с.

Additional literature

7. List of web resources that are necessary for the course (training module) mastering

1. http://www.opennet.ru/man.shtml
2. http://www.pcmag.ru/encyclopedia/search.php
3. http://www.mcst.ru/publik_old.shtml

Basic literature:

2. Robert Love. Linux Kernel Development. – Indianapolis: Novell Press, 2005. (Русский перевод: Лав, Роберт. Разработка ядра Linux. – М: Вильямс, 2008.)

3. Семенихин С.В., Ревякин В.А. и др. ОС Linux и режим реального времени. – «Вопросы радиоэлектроники», серия ЭВТ, Выпуск 3, 2009.

4. Федоров, А. В. Оптимизация библиотеки нитей NPTL в составе ОС Linux для систем жесткого реального времени. - «Программирование», 2011, №4.

Additional literature:

1. Uresh Vahalia. UNIX Internals: The New Frontiers. (Русский перевод: Вахалия, Юреш. Unix изнутри. – Спб: Питер, 2003.) – Доступно для скачивания в сети.

8. List of information technologies used for implementation of the educational process, including a list of software and information reference systems (if necessary)

- Installed documentation system on UNIX (man and info system with the necessary help pages);
- C language compilers (gcc, clang);
- tool for viewing presentations, tests: libreoffice package;
- means for viewing pdf documents.

9. Guidelines for students to master the course

A student studying a discipline must, on the one hand, master the general conceptual apparatus, and on the other hand, must learn to put theoretical knowledge into practice.

As a result of studying the discipline, the student must know the basic definitions, concepts.

Successful development of the course requires intense independent work of the student. The course program provides the minimum necessary time for the student to work on the topic. Independent work includes:

- reading and taking notes of recommended literature;
- study of educational material (according to lecture notes, educational and scientific literature);
- preparation for differentiated classification.

The management and control of the student's independent work is carried out in the form of individual consultations.

It is important to gain an understanding of the material being studied, and not its mechanical memorization. If it is difficult to study certain topics, questions, you should consult a teacher.

Assessment funds for course (training module)

major:	Information Science and Computer Engineering
specialization:	Computer Science/Информатика
	Phystech School of Applied Mathematics and Informatics
	Chair of Algorithms and Programming Technologies
term:	2
qualification:	Bachelor

Semester, form of interim assessment: 4 (spring) - Grading test

Author: V.V. Yakovlev, candidate of physics and mathematical sciences

1. Competencies formed during the process of studying the course

Code and the name of the competence	Competency indicators
Gen.Pro.C-1 Apply fundamental knowledge acquired in the physical and mathematical fields and/or natural sciences and use it in professional settings	Gen.Pro.C-1.1 Analyze the task in hand, outline the ways to complete it

2. Competency assessment indicators

As a result of studying the course the student should:

know:
- Main types of objects of operating systems;
- methods for implementing objects of operating systems;
- structure of operating systems using Linux as an example;
- means of parallelization of calculations of the POSIX standard;
- methods of real-time implementation in the OS;
- methods of organizing OS design work;
- testing methods for large software systems.

be able to:
- Use the basic mechanisms of OS such as Linux;
- synchronize parallel computing.

master:
- Basic mechanisms for working with large software systems.

3. List of typical control tasks used to evaluate knowledge and skills

Not provided.

4. Evaluation criteria

The list of control questions for passing the test:

1) The relationship of hardware, operating system and programming system in a computer system.
2) Types of objects in computing systems (types implemented by hardware, operating system, user programs).
3) Methods for implementing types (type control): dynamic control, control at compile time. Hardware requirements for efficient type implementation.
4) Pointers to objects, virtual memory and the problem of stuck pointers.
5) Exceptions and abnormal methods for completing procedures.
6) The main types of objects sold by operating systems
7) The main properties of the address space and its relationship with other main objects of the operating system.
8) The main properties of the executable program and its relationship with other main objects of the operating system.
9) The main properties of the library of functions and its relationship with other main objects of the operating system.
10) The main properties of the computational flow (procedure call stack) and its relationship with other main objects of the operating system.
11) The main properties of the file and file system and their relationship with other main objects of the operating system
12) What is a user, task, session. By what means does the OS implement these concepts.
13) The approach of programming languages to data protection. Contextual naming of data arrays.
14) Organization of a file system for the contextual naming of external objects (files).

15) Types of data exchanges between files and arrays in operational (virtual) memory.

16) The Unix process as a connection in one design of the address space, the executable program and the stack of procedure calls. Address space expansion by file open table.

17) Signals, process groups, means of working with signals in Unix-type OS.

18) The main attributes of the process, generation and termination of the process, signals and work with them, synchronization of processes through signals in Unix-type OS.

19) Sessions, tasks (work) in OS type Unix. Teams of work with tasks.

20) File system, directories, file naming. Managing file lifetimes on an Unix-type OS.

21) Files and address space in a Unix-type OS.

22) Data exchange between files and RAM: basic buffered and streamed buffered; direct exchange; asynchronous exchange. Inter-process communication via pipe files in a Unix-type OS

23) Users and user groups. Data protection through file attributes on a Unix-type OS.

24) Structured use of non-local transitions in C.

25) Command language, options and arguments, assigning file names, handling symbolic links. Basic utilities like Unix.

26) Flags of the compiler and flags of the link editor. How to create a dynamically loaded library and how to load such a library with your program. What is dlopen ().

excellent

10 comprehensive, systematized, deep knowledge of the curriculum of the discipline and the ability to confidently apply them in practice when solving specific problems, free and correct justification of decisions made;

9 systematic, deep knowledge of the curriculum of the discipline and the ability to confidently apply them in practice when solving specific problems, the correct justification of decisions made;

8 deep knowledge of the curriculum of the discipline and the ability to apply them in practice when solving specific problems, the correct justification of decisions made;

good

7 firmly knows the material, correctly and essentially sets out it, knows how to apply the knowledge gained in practice, but admits some inaccuracies in the answer or in solving problems;

6 knows the material, correctly presents it, knows how to apply the acquired knowledge in practice, but admits some inaccuracies in the answer or in solving problems;

5 knows the basic material, correctly presents it, knows how to apply the knowledge gained in practice, but admits inaccuracy in the answer or in solving problems;

satisfactorily

4 fragmented, fragmented nature of knowledge, insufficiently correct wording of basic concepts, violation of logical sequence in the presentation of program material, but at the same time he owns the main sections of the curriculum necessary for further training and can apply the acquired knowledge in the standard situation;

3 the nature of knowledge is sufficient for further training and can apply the acquired knowledge on the model in a standard situation;

unsatisfactory

2 does not know most of the main content of the curriculum of the discipline, makes gross errors in the wording of the basic concepts of the discipline and does not know how to correctly use the knowledge gained in solving typical practical problems.

1 does not know the wording of the basic concepts of the discipline and does not know how to use the knowledge gained in solving typical practical problems.

5. Methodological materials defining the procedures for the assessment of knowledge, skills, abilities and/or experience

The test time is 2 academic hours.During the test, students can use the discipline program and the source code.