SOFTWARE OF HIGH PERFORMANCE COMPUTER SYSTEMS (Silabus)

Details of the academic discipline

Cycle of Higher Education	***First cycle of higher education (Bachelor's degree)***
Field of Study	*12 Information Technologies*
Speciality	*121 Software Engineering*
Education Program	*Software engineering of computer systems*
Type of Course	*Normative*
Mode of Studies	*full-time*
Year of studies, semester	*3nd year (5rd semester)*
ECTS workload	*4 credits (ECTS). 120 hours,*
Testing and assessment	*Eksam, MCW*
Course Schedule	*http://rozklad.kpi.ua/*
Language of Instruction	*English*
Course Instruction	*Lecturer: PhD, Associate Professor, Olexandr Korochkin, mobile +380508203861 email avcora@gmail.com* *Teacher of laboratory work: PhD, Associate Professor, Olexandr Korochkin, mobile +380508203861 email avcora@gmail.com*
Access to the course	*BigBlueButton, Google, Zoom, http://comsys.kpi.ua*

Program of educational discipline

1. Description of the educational discipline, its purpose, subject of study and learning outcomes

The discipline "Software of high performanece computer systems" is aimed at students mastering the methods and means of building parallel programs. Knowledge of the basics of parallel programming necessary for creating software for parallel and distributed computer systems, real-time systems, Internet applications, mobile devices.

The discipline provides the following program learning outcomes of the educational and professional program Computer systems and networks: FK17, FK18, PRN25, PRN27.

The purpose of teaching the discipline "Parallel Programming" is to provide students with knowledge and skills in the field of creating software for parallel computer systems, in particular the concept of flow, implementation of flows in modern parallel programming languages and libraries, methods and means of organizing the interaction of flows, solving the problem of mutual exclusion and synchronization of flows, tools for programming interaction of flows: semaphores, mutexes, events, critical sections, monitors, messages, analysis and construction of parallel algorithms. The purpose of the educational discipline is to develop students' competence in developing software for computer systems with parallel or distributed architecture based on mastery of tools of modern languages and libraries of parallel programming at workplaces and units during future professional activities in the first position. According to the results of studying the discipline, the student should be able to solve professional tasks and possess the following:: evaluate the features of the applied PCS; use methods of creation and evaluation of parallel

algorithms; to use tools of modern languages and parallel programming libraries to create software for modern PCs, to master the basic methods of creating programs based on the concept of processes.

ABILITY:

- development of software for computer systems with parallel or distributed architecture based on knowledge of modern languages and parallel programming libraries

- analyze the structure of a parallel computer system

- analyze the parallel properties of the problem being solved

-develop and evaluate parallel algorithms

- organize calculations in parallel and distributed computer systems

KNOWLEDGE:

-methods and means of organizing calculations in high-performance computer systems

-structures of modern parallel computer systems, -organization of memory and communication of processors,

-methods of analysis, evaluation and presentation of parallel algorithms, concepts of development of parallel programs

-methods of process programming, the concept of flow, operations with flows, -methods and means of organizing the interaction of flows,

-process programming tools using modern languages and parallel programming libraries,

-process interaction models,

- resolutions and solutions to the task of mutual exclusion and synchronization of processes,

- methods of setting up parallel programs.

***SKILLS**:*

- organization of computing processes in modern computer systems,

- create software for high-performance computer systems with different architectures,

- build a parallel algorithm,

- analyze the effectiveness of the parallel algorithm, form process algorithms, - program processes and create a parallel program,

- organize effective interaction of processes depending on the structure of the PCS,

- place processes on PCS processors,

-debug and run a parallel program/

2. Pre-requisites and post-requisites of the discipline (place in the structural and logical scheme of training according to the relevant educational program)

Necessary disciplines: "Programming", "Object-oriented programming", "System programming", "Data structures and algorithms", "Software engineering", "Algorithms and calculation methods"

Disciplines based on the learning outcomes of this discipline: "System software", "Computer systems"

1. Content of the academic discipline

Chapter 1. Structures of parallel computer systems (PCS)

Topic 1.1 Basic structures of PCS

Topic 1.2 Organization of memory and connections in PCS

Chapter 2. Programming of parallel calculations

Topic 2.1 Peculiarities of parallel computing programming.

Topic 2.2 Life cycle of development of parallel programs.

Chapter 3. Analysis and construction of parallel algorithms Topic 3.1 Concept of unlimited parallelism

Topic 3.2 Tier-parallel form of representation of a parallel algorithm. Brent's lemma. The Munro-Peterson theorem.

Topic 3.4 Parallel algorithms for solving the main problems of linear algebra

Chapter 4. Parallel processes (threads)

Topic 4.1 Concept of process. Process status. Operations with processes

Topic 4.2 Processes in modern parallel programming languages and libraries Chapter 5. Communication of processes
Topic 5.1 Types of process communication Topic 5.2 Process interaction models.
Chapter 6. Problems of mutual exclusion and synchronization
Topic 6.1 Statement and general scheme of solving the problem of mutual exclusion and synchronization
Topic 6.2 Dekker's algorithms Topic 6.3 Semaphores.
Topic 6.4 Mutexes.
Topic 6.5 Events.
Topic 6.6 Critical sections. Chapter 7. Monitors
Topic 7.1 Concept of monitors
Topic 7.2 Implementation and use of monitors
Chapter 8. Interaction of processes through message links. Topic 8.1 General concept of message mechanism
Topic 8.2 Peculiarities of implementation and use of the message mechanism.

Guidelines

The method of studying the discipline requires the use of modern hardware and software, which is connected with parallel and distributed processing. As for the hardware component, parallel computer systems with multi-core architecture should be used in the first place. Attention should be paid to the structural organization of modern multi-core processors, the organization of multi-level cache memory and the communication of cores. Laboratory work should be done in a classroom equipped with such multi-core computers.

Modern software tools that allow programming for parallel (distributed) computer systems are based on the use of parallel programming languages and libraries. These include Java, C#, and Ada languages. OpenMP, MPI, PVM. It is necessary to consider all such systems, because they differ both in terms of the concept of creating processes and in the means of organizing the interaction of processes.

The Ada language was chosen as the base language, which supports the classic approach to creating processes through special task modules, and also implements both models of process interaction organization through shared variables and message links. In addition, Ada language resources are widely available on the Internet.

3. Educational materials and resources

Basic:

1. *Loutsky G., Zhukov I., Korochkin A. Parallel Computing. – Kyiv, Kornechuk, 2007. -216 pp. The handle is provided by the Ministry of Education and Culture //comsys.kpi.ua*
2. *Multicore programming in the Ada language: English [Electronic resource]: teaching. manual for students specializations 123 "Computer engineering", specializations "Computer systems and networks", "Programming technologies for computer systems and networks" O.V. Korochkin; KPI named after Igor Sikorsky. – Electronic text data (1 file: 2.44 Mbyte). – Kyiv: KPI Igor Sikorskyi, 2018. – 114 p. The vulture is provided by the Methodical Council KPI named after Igor Sikorsky.*
3. *Software of high performance computer systems. Laboratory practice. /Korochkin O.V., - Kyiv: KPI named after Igor Sikorskyi, 2022. – 24 p. Electronic resource. The fretboard was provided by the Academic Council of the FIOT KPI Igor Sikorskyi (protocol No. 11 dated July 11, 2022)*

Additional:

4. *Ringler R. C# Multithreaded and Parallel Programming. Birmingham: Packt Publishing, 2014. – 323с. ([Електронний ресурс] - Режим доступу: https://www.pdfdrive.com/c-multithreaded-and-parallel-programming-develop-powerful-c-applications-to-take-advantage-of-todays-*

multicore-hardware-e187077213.html

5. *Terrell R. Concurrency in .NET. Shelter Island: Manning, 2018. – 534p. https://www.pdfdrive.com/concurrency-in-net-modern-patterns-of-concurren...pdf*

Educational content

6. Methodology

Names of sections and topics	Hours				
	Total	Lecturesï	Practics	Labs	SRS
1	2	3	4	5	6
Chapter 1. Sructures of parallel computer systems(PCS)					
Topic 1.1 Base structures	*1,5*	*1*	*-*	*-*	*0,5*
Topic 1.2 Memory organization	*1,5*	*1*	*-*	*-*	*0,5*
Total by chapter 1	***3***	***2***	***-***	***-***	***1***
Chapter 2. Programming of parallel calculation					
Topic 2.1 Features parallel computing programming.	*1,5*	*1*	*-*	*-*	*0,5*
Topic 2.2 Life cycle development of parallel programs	*1,5*	*1*	*-*	*-*	*0,5*
Total by chapter 2	***3***	***2***	***-***	***-***	***1***
Chapter 3. Analysis and construction of parallel algorithms					
Topic 3.1 The concept of unlimited parallelism	*1,5*	*1*	*-*	*-*	*0,5*
Topic 3.2 Tier-parallel a form of representation of a parallel algorithm	*1,5*	*1*	*-*	*-*	*0,5*
Topic 3.3 Parallel algorithms solutions of basic problems of linear algebra	*3*	*2*	*-*	*-*	*1*
Total by chapter 3	***6***	***4***			***2***
Chapter 4. Parallel processes (threads)					
Topic 4.1 Concept of the process. State process Operations with processes	*3*	*2*			*1*
Topic 4.2 Processes in modern parallel languages and libraries programming	*10*	*4*	*-*	*2*	*4*
Total by chapter 4	***13***	***6***	***-***	***2***	***5***
Chapter 5. Communication of processes					
Topic 5.1 Types of communication processes	*6*	*1*	*-*	*-*	*5*

Topic 5.2 Interaction models processes.	*4*	*1*	*-*	*-*	*3*
Total by chapter 5	***10***	***2***	***-***	***-***	***8***
Chapter 6. Problem of mutual exclusion and synchronization					
Topic 6.1 Formulation and general scheme of solving the mutual exclusion problem and synchronization	*1,5*	*1*			*0.5*
Topic 6.2 Dekker's algorithms	*1,5*	*1*	*-*	*-*	*0.5*
Topic 6.3 Semaphores	*1,5*	*2*	*-*		*1*
Topic 6.4 Mutexes	*1,5*	*1*			*0,5*
Topic 6.5 Events	*1,5*	*1*	*-*		*1*
Topic 6.6 C ritical sections	*3*	*2*	*-*		*1*
Total by chapter 6	***12,5***	***8***	*-*		***4,5***
Chapter 7. Monitors					
Topic 7.1 Concept of monitors	*3*	*2*	*-*	*-*	*1*
Topic 7.2 Implementation and use of monitors	*3*	*2*	*-*		*1*
Total by chapter 7	***6***	***4***	*-*		***2***
Chapter 8. Interaction of processes through messages					
Topic 8.1 General concept message mechanism	*1,5*	*1*	*-*	*-*	*0,5*
Topic 8.2 Features of implementation message mechanism	*1,5*	*1*	*-*		*0,5*
Разом за розділом 8	***3***	***2***	*-*		***1***
Chapter 9. Development of programs for PCS SM and PCS LM					
Topic 9.1 Development of programs for PCS SP	*28,5*	*4*		*12*	*12*
Topic 9.2 Development of programs for PCS LP	*10*	*2*		*4*	*4*
Total by chapter 9	***38,5***	***6***		*16*	***16,5***
MCW	***1***				***1***
Exam	***30***				***30***
In total:	***120***	***36***	***-***	***18***	***66***

Lectures

№ lectures	*The name of the topic of the lecture and a list of main questions (a list of didactic tools, references to the literature and tasks on the SRS)*
1	***Structures of parallel computer systems (PCS)*** *Basic structures of the PCS. Organization of memory and connections in the PCS. PC with local memory. PCS with shared memory [1, p. 10-17].* *SRS: Perform calculations of the topological characteristics of the PCS LP (linear, ring, star, lattice, hypercube) IWS [1, p. 21].*
2	***Programming of parallel calculations*** *Peculiarities of the task of parallel programming: research on parallelism tasks, development of parallel algorithms, programming of parallel processes, interaction of processes, solution of the problem of mutual exclusion, synchronization of processes, placement of processes on processors, planning of execution of processes, debugging and execution [1, p.37-38.]* *Features of the development life cycle of programs for parallel systems. Stages of development: analysis of requirements, design, implementation, testing, production, modification, support [1, c.38-48].* *IWS: Perform a software development life cycle analysis for the search operation of the maximum element of the matrix .[1, p. 66-67].*
3	***3 Analysis and construction of parallel algorithms*** *The concept of unlimited parallelism. Purpose and application [1, c.38-44]. Tier-parallel form (YPF) of the representation of a parallel algorithm.* *Definition and purpose of the JPF. Rules for the construction of the JPF. YPF parameters: rank, width, height. Application. [1, c.39-40]. The Munro-Peterson theorem. Brent's Lemma to establish the connection between bounded and unbounded parallelism Appointment. Application examples. The Munro-Peterson theorem for evaluating arithmetic expressions. Application.[1, c. 39-43].* *IWS: Build a parallel algorithm for a given expression, determine To, Tp. Ku [1, c. 76-77].*
4	***Parallel algorithms for solving the main problems of linear algebra*** *Parallel algorithms for operations on vectors and matrices. Sorting. Finding the maximum element. Solutions of systems of algebraic equations [1, c. 42-45].* *IWS: Develop parallel algorithms for given vector-matrix operations [1, c. 76-77].*

5	***Process concept.*** *Types of process interaction. Definition of the process. Types of process status. Process operations. Types of processes. [1, c.26-30]* *Representation of processes in the OS. Process control unit. Process and resource are fundamental concepts of modern operating systems. Process control in the OS. Process control unit. [1, c.27].* *Types of process interaction. Interaction of processes. Process communication. Synchronization of processes[1, c.78-79]* *IWS: Perform an analysis of various approaches to programming processes (flows) in modern parallel programming languages and libraries [1, c. 26-54].*
6	***Processes in modern programming languages.*** *The language Adal. Java language. Language C# [1, c.* *26-55]. SRS: Flow programming in the Python language [1, p. 50]*
7	***Processes in special parallel programming libraries.*** *Win32 library. MPI library. PVM library. OpenMP library [1, c.26-55].* *IWS: Flow programming in the PVM library [5, p. 30]*
8	***Types of process synchronization and interaction*** *Tasks of process interaction: synchronization and data exchange. Peculiarities of synchronization mechanisms. [1, c. 50-54]* *SRS: Peculiarities of interaction of processes [1, c.28-29]*
9	***Classical problems of interaction of processes.*** *Mutual exclusion. Producer-Consumer. Reader - Writer. Philosophers dining. [Addendum 3, c. 124-130].* *IWS: Task Factory-Store [1, c.76-79]*
10	***Statement and general scheme of the solution of the mutual exclusion problem*** *Statement of the problem. General scheme of the decision. Critical areas. ENTRY and EXIT structures. [1, c.78-82]..* *SRS: Decker's Algorithms 2 - 4 [Additional 3, c. 62-63].*
11	***11 Semaphores*** *The general concept of semaphores. Semaphore type. Operations with semaphores. Use of semaphores. Realization. Examples. [1, c. 91-100].* *IWS: Implementation and application of multiple semaphores*
12	***Mutexes*** *General concept of mutexes. Operations with mutexes. Use of mutexes.* *Realization. Examples. [1, c. 99, 126].* *SRS: Implementation and use of mutexes in the Python language [16, c. 131-167].*
13	***1Events*** *General concept of events. Operations with fields. Application of events. Realization. xamples. [1, c. 137-142].* *IWS: Implementation and application of events in the Python language [3, c. 131-167].*

14	***Critical sections*** *Concept of critical sections. Application. Implementation in C# language and Win32 library [1, c.100-101]. Distributed variables. Constructions shared, atomic, volatile. . [1, c. 82-90].* *IWS: Implementation of the mechanism of critical sections in the OpenMP library [1, c. 127**]***
15	***The concept of monitors*** *The general concept of monitors. Data and procedures. Features of procedures. A solution to the problem of mutual exclusion and synchronization. [1, c.102-104].* *SRS: Comparison of implementation[1, c.76-79]*
16	***Implementation of monitors.*** *Ada Language: Protected Modules. Monitors in Java. Application examples. [1, c. 104-110] IWS: Task Factory-Store [1, c.76-79]*
16	***General concept of the message mechanism*** *Send and Receive operations. Blocked and non-blocking data transfer operations. [1, c.152-177]* *IWS: types of transfer operations in the MRI library [1, c.168-177]*
17	***Implementation of the message mechanism*** *The rendezvous mechanism in Ada's and Occam's languages. MPI Library [2, c.153-176]* *IWS: PVM Library [1, 162-177] , [5, 117-122]*
18	***Chapter 9. Development of programs for PCS SP and PCS LP*** *Development of programs for PCS JV. [1, c.153-175]* *Development of programs for PCS LP [1, c.176-185]* *IWS: PVM Library [1, 162-177] , [4, 117-122]*

Laboratory works

The main tasks of the cycle of laboratory classes are for students to acquire the necessary practical skills for developing parallel programs based on the concept of processes, using parallel programming languages and libraries Java, Ada, C#, WinAPI, OpenMP, MPI on the basis of modern computer equipment in the form of powerful multi-core systems

№ з/п	*The name of the laboratory work*	*Hours*
1	*Development of parallel programs using semaphores and mutexes*	*4*
2	Development of parallel programs using events and critical sessions	*4*
3	Development of parallel programs for the use of monitors	*4*
4	Development of parallel programs using the OpenMP library	*2*
5	*Development of parallel programs for the application of domestication*	*4*
	In total:	*18*

7.Self-study

№	The name of the topic submitted for independent processing	Hours
1	Transcomputer systems. Hardware and software. Occam's language [1]	4
2	Modern distributed (cluster) systems. Structural organization. Software [www.top500.com]	4
3	PVM library. Flow programming. Organization of interaction threads [4]	4
	Total	12

Politic and control

The following factors are taken into account when enrolling and evaluating laboratory works:
• Complete completion of the task for laboratory work according to the individual option;
• Timeliness of laboratory work according to the schedule;
• Independent performance of laboratory work and absence of signs of plagiarism;
• Answers to questions about the content of laboratory work during its defense. When evaluating control works, the following are taken into account:
• Correctness and completeness of tasks;
• Number of completed tasks under limited time conditions;
• Independent performance of tasks and absence of signs of plagiarism;
• The number of test execution attempts that precede the one being evaluated.
To prepare for the tests, students receive a list of theoretical questions and the content of typical problems that will be in the tests.
During the first and second attestation, the number of laboratory works and control works enrolled at the time of the attestation is taken into account.

8. Monitoring and grading policy

The Rating system of evaluation (RSE) in the discipline, the semester control of which is provided in the form of a credit, for full-time education is developed according to the RSE-1 type and includes the evaluation of measures of current control in the discipline during the semester.

The rating evaluation of the acquirer consists of the points received by the acquirer based on the results of current control measures, incentive and penalty points.

Applicants who have fulfilled all the conditions for admission to the test and have a rating of 60 or more points receive a rating corresponding to the rating without additional tests.

With students who have fulfilled all the admission requirements and have a rating score of less than 60 points, as well as with those students who wish to improve their rating score, the teacher conducts a semester control in the form of a credit test or interviews

Types of control from the educational discipline include:

Laboratory works:

Independent performance of three to five laboratory works (optional) is planned. The topics of laboratory works are coordinated in time and content with the topics of lectures. Carrying out laboratory work in its entirety allows you to acquire practical skills in thread programming using modern parallel programming languages and libraries.

Current control:

It is planned to carry out modular control work (MCW)

Semester control

Assessment is conducted in the form of an interview with the student to objectively determine the level of knowledge, skills and practical skills acquired during the semester.

Table 1.Table 1

Assessment of individual types of student academic work (in points)

Type of educational work		**Total by type of work**
Laboratory work № 1		10-18
Laboratory work № 2		10-18
Laboratory work № 3		10-18
Laboratory work № 4		10-18
Laboratory work № 5		10-18
In total	**Rl**	50-90
Modular control work (MCW)	**Rk**	10
In total	**Ro= Rl + Rк**	**60-100**
Assessment in the form of an interview with the student (optional)		**20**
In total (**R = Ro + Re**)		**100**

The student's individual current rating (Rп) consists of the points he receives for performing laboratory work and MKW. During the semester, students perform a selected number (5) of laboratory work. The maximum number of points for each laboratory work is 16. Points are awarded for:

- theoretical component – (5-9) points,
- practical component - (5-9) points.

The maximum possible score for laboratory work is 10 points. The maximum number for all laboratory works is 5x18=90 points.

Calculation of the scale size (R) of the rating.

The sum of the weighted points of control measures during the semester is:

Ro= Rl + Rk, where Rp is the student's semester rating (MCW, laboratory works).

A necessary condition for a student's admission to credit is his individual semester rating (Ro), not less than 20 points, and the absence of arrears from laboratory work and MCW. If the mentioned requirements are not met, the student will not be admitted to the credit.

The final performance score or the results of the Fail/ Pass Test are adopted by university 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

7. Added informations

As part of the study of the discipline it is allowed to credit the points obtained as a result of distance courses on the "Coursera" platform (5 credits), provided that the program of the given course has been previously agreed with the teacher and provided that an official certificate has been obtained.

Syllabus of the course

Is designed by teacher PhD, Associate Professor, Olexandr Korochkin

Adopted by Department of Computing Technics (protocol № 10, 25 May 2023)

Approved by the Faculty Board of Methodology (protocol № 11, 30 June 2023)

,,