INFORMATICS. PART 1. FUNDAMENTALS OF PROGRAMMING AND ALGORITHMS

Work program of the academic discipline (Syllabus)

Course details

Level of higher education	***First (bachelor's)***
Field	*17 — Electronics, Automation, and Electronic Communications*
Specialty	*172 — Electronic Communications and Radio Engineering*
Educational program	*Intelligent technologies of radio-electronic engineering* *Information and communication radio engineering Radio engineering computerized systems* *Radio Electronic Warfare Technologies*
Status of the discipline	*Regulatory*
Form of study	*Full-time (day)*
Year of study, semester	*1st year, fall semester*
Scope of the discipline	*6 credits* (*Lectures: 18 hours; Labs: 72 hours; Independent work: 90 hours*)
Semester control/control measures	*Exam*
lesson schedule	*https://schedule.kpi.ua/*
Language of instruction	*Ukrainian*
Information about the course leader/teachers	Lecturer: *Serhii Vyshnevyi, Ph.D. s.vyshnevyi@kpi.ua* Laboratory: *Pavlo Yuriyovych Katin, PhD, Associate Professor,* *Serhiy Valeriyovych Vyshnevyi, PhD, Igor Olegovych Tovkach*, PhD *Romanenko Taras Volodymyrovych*
Course placement	https://do.ipo.kpi.ua/course/view.php?id=6264

Curriculum

1. Description of the course, its purpose, subject matter, and learning outcomes

The educational component "Computer Science. Part 1. Fundamentals of Programming and Algorithms" involves studying the C programming language, which is mastered by implementing appropriate algorithms as part of educational tasks. This approach allows students to develop the following skills: formalizing a given task, developing algorithms for solving the task, implementing algorithms by writing program code, detecting errors, and debugging the program. The choice to study the C programming language is dictated by the following: the C programming language is widely used in practice, including for programming microcontrollers (microcontroller programming is studied in relevant professional training disciplines and elective disciplines), thus, mastering this discipline is the basis for studying the following disciplines related to the study of the corresponding programming languages and the study of microcontroller programming. The skills acquired will be necessary for performing tasks involving mathematical modeling, statistical modeling, microcontroller device design, radio engineering system development, etc., as well as for independent study of issues related to programming in the procedural paradigm. Considering that a number of modern programming languages have C-like syntax, successful mastery of the material in this discipline will greatly facilitate the process of independent study or study within the framework of relevant disciplines (which are available for selection) of programming languages that involve writing programs in an object-oriented paradigm.

The aim of the discipline is to study the C programming language at a level sufficient to implement methods of data analysis and processing, to master the relevant common and widely used algorithms and data structures that are used to solve a wide range of tasks, including those that can be applied in the implementation of programs for signal and data processing by means of electronic computing technology, in embedded systems, etc.

Subject of study: the C programming language, data processing algorithms, and data structures.

In accordance with the educational and professional program, the discipline provides the following general (GC) and professional competencies (PC):

General competencies (GC)
GC 02 Ability to apply knowledge in practical situations.
GC 04 Knowledge and understanding of the subject area and understanding of professional activity.
SS 07 Ability to learn and acquire modern knowledge.
HS 08 Ability to identify, pose, and solve problems.

Professional competencies (PC)
PC 02 Ability to solve standard professional tasks based on information and bibliographic culture using information and communication technologies and taking into account the basic requirements of information security.
PC 04 Ability to perform computer modeling of devices, systems, and processes using universal application software packages.

Program learning outcomes (**PLO**):
PRO 18 Find, evaluate, and use information from various sources necessary for solving professional tasks, including reproducing information through electronic search.

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

The educational component "Computer Science" is taught in the 1st semester of the 1st year of study for students enrolled in the educational program "Intelligent Technologies of Radio Electronics." The competencies acquired by students in the process of studying this discipline are applied in mastering the discipline "Programmable Tools in Intelligent Radio Electronics."

Prerequisites for studying this discipline are "General Physics" and "Analytical Geometry and Linear Algebra."

3. Course content

Topic 1. Number systems: Binary, octal, and hexadecimal number systems.
Topic 2. Straight and complementary code. Fixed-point format. Floating-point format.
Topic 3. Program structure in C. Data types. Operations.
Topic 4. Conditional operators.
Topic 5. Pointers. Basics of working with functions.
Topic 6. One-dimensional and multidimensional static arrays.
Topic 7. One-dimensional and multidimensional dynamic arrays.
Topic 8. Working with files. Character strings and functions for working with character strings.
Topic 9. Structures. Basics of working with structures.

4. Teaching materials and resources

Basic literature.

1. Shpak Z.Y. Programming in C: Textbook / Z.Y. Shpak. — L.: Oriana-Nova. — 2006. — 432 p.
2. Lyubashenko, N.D. Programming-2. Language C. Lecture notes / N.D. Lyubashenko. — Kyiv: Igor Sikorsky Kyiv Polytechnic Institute. — 2019. — 144 p.
3. Tatarchuk D.D. Informatics: textbook / D.D. Tatarchuk, Yu.V. Didenko, A.S. Franchuk. — Kyiv: NTUU "KPI", 2015. — 199 p.
4. Computer Science. Fundamentals of Programming and Algorithms: Programming Language C. Laboratory Practicum [Electronic resource]: textbook / comp.: S.V. Vyshnevyi, P.Yu. Katin, Ye.V. Krylov. — Electronic text data (1 file: 3.21 MB). — Kyiv: Igor Sikorsky Kyiv Polytechnic Institute, 2022. — 221 p. (https://ela.kpi.ua/handle/123456789/48158).
5. Vyshnevyi, S.V. Informatics. Part 1. Fundamentals of Programming and Algorithms. Lecture Course [Electronic resource]: textbook for students majoring in 172 Electronic Communications and Radio Engineering / S.V. Vyshnevyi; Igor Sikorsky Kyiv Polytechnic Institute. – Electronic text data (1 file: 4.52 MB). – Kyiv: Igor Sikorsky Kyiv Polytechnic Institute, 2023. – 315 p. (https://ela.kpi.ua/handle/123456789/57429)
6. Vyshnevyi, S. V. Informatics. Part 1. Fundamentals of programming and algorithms. Home assignment [Electronic resource]: textbook for students majoring in 172 "Electronic Communications and Radio Engineering" / S. V. Vyshnevyi; Igor Sikorsky Kyiv Polytechnic Institute. – Electronic text data (1 file: 2.2 MB). – Kyiv: Igor Sikorsky Kyiv Polytechnic Institute, 2023. – 157 p. (https://ela.kpi.ua/handle/123456789/57828)

7. Tatarchuk, D.D. Programming in C and C++: a textbook / D.D. Tatarchuk, Yu.V. Didenko. — Kyiv, 2012. — 112 p.

Additional literature

8. Functional and logical design: Combinational devices [Electronic resource]: textbook for students majoring in 172 "Telecommunications and Radio Engineering," specializing in "Information and Computing Tools for Electronic Systems" / Igor Sikorsky KPI; compiled by: A.Yu. Varfolomeev. – Electronic text data (1 file). – K.: Igor Sikorsky KPI, 2017. – 135 p.

9. Chavan S. C Recipes: A problem-solution approach / S. Chavan. — Apress, 2017. — 464 p.

10. Mochurad L.I. S. Fundamentals of Programming. Theory and Practice: Textbook / L.I. Mochurad, N.I. Boiko. — L.: Galich–Press, 2019. — 150 p.

11. Plantz R.G. Introduction to computer organization / R.G. Plantz // Sonoma State University. — Access mode: https://bob.cs.sonoma.edu/IntroCompOrg-RPi/intro-co-rpi.html — Date of access: 01.06.2024. — Title from the screen.

12. IEEE Standard 754 Floating Point Number. — Access mode: https://www.geeksforgeeks.org/ieee-standard-754-floating-point-numbers/?ref=lbp — Date of access: 01.06.2024. — Title from the screen.

Website for downloading files for installing the development environment on a personal computer CodeBlocks: https://www.codeblocks.org/downloads/binaries/

Educational content

5. Methodology for mastering the academic discipline (educational component)

The educational component "Computer Science. Part 1. Fundamentals of Programming" involves lectures and practical (laboratory) lessons.

Lectures

Topic 1. Number systems. Binary, octal, and hexadecimal number systems.

Subject and content of the discipline. Number systems. Conversion of decimal numbers to binary, octal, and hexadecimal number systems. Reverse conversion. Methods for quickly converting binary code to octal or hexadecimal code. Reverse conversion. Introduction to the universal software package (integrated development environment) Code::Blocks, creating a project in the Code:Blocks software environment.

Assignments for independent study. Installing and familiarizing yourself with the interface of the Code::Blocks integrated development environment. Using the binary number system for data storage in computer and digital technology.

Topic 2. *Direct and complementary code. Fixed-point format. Floating-point format.*

Representation of unsigned and signed integer data types in computer memory. Fixed-point format for storing integer data types. Representation of real data types in computer memory. Floating-point format. Method for converting a real number containing an integer and fractional part to the binary number system.

Assignments for independent study. Binary arithmetic. Performing the arithmetic operation of addition using the binary number system. IEEE 754 standard for storing single-precision real numbers in computer memory.

Topic 3. *Program structure in the C language. Data types. Operations.*

Stages of processing source code to obtain a file intended for execution. Program structure in the C programming language. Rules for compiling identifiers. Purpose of header files. Identifiers and service words. Basic data types. Declaration of variables of basic data types. Declaration of constants using the #define directive. C language operations. Rank and associativity of operations.

Assignments for independent work. Basic data types: range of values and size in bytes. Bitwise operations. The sizeof operation. The rule of automatic conversion of data types in arithmetic expressions.

Topic 4. *Conditional statements.*

Conditional selection statements if, if/else. Syntax and structural diagram of conditional selection statements. Loop statements for, while, do/while. Syntax and structural diagram of loop statements. Using unconditional statements break and continue together with loop statements. Examples of using conditional statements.

Assignments for independent study. Conditional switch operator. Syntax and structural diagram of the switch operator. Infinite loop. Exiting an infinite loop. Unconditional goto operator. Using the goto operator to perform cyclic actions.

Topic 5. *Pointers. Basics of working with functions.*

Declaration and initialization of a pointer. NULL pointer. Accessing a memory cell whose address is stored in a pointer. Pointer to void. Functions that return a result and functions of type void. Function prototype. Function description. Formal and actual parameters. Standard library functions. Passing parameters to a function by value and by reference.

Assignments for independent work. Type conversion operation applied to pointers. Pointer to pointer. Pointer to function. Passing arguments to the main () function. Recursive function calls.

Topic 6. ***One-dimensional and multidimensional static arrays.***

One-dimensional static arrays. Initialization of one-dimensional static arrays. Indexing elements of a one-dimensional array. Array name as a pointer to the first element. Placement of array elements in memory. Passing a one-dimensional array to a function as a parameter. Multidimensional arrays. Two-dimensional array and its placement in memory. Passing a two-dimensional array to a function.

Assignment for independent study. Declaration, initialization, and display of multidimensional arrays (using three-dimensional, four-dimensional, and five-dimensional arrays as examples). Determining the number of elements in a static array using the size of operation.

Topic 7. ***One-dimensional and multidimensional dynamic arrays.***

Dynamic memory. Dynamic one-dimensional array. Memory allocation functions for dynamic arrays. Accessing elements of a one-dimensional dynamic array using a pointer. Two-dimensional dynamic array. Memory layout of elements of a two-dimensional dynamic array. Passing dynamic arrays to a function. Processing data from dynamic arrays. Examples of working with dynamic arrays: using arrays to model a random process and evaluate its statistical parameters by writing a program using the Code::Blocks universal software development kit.

Assignments for independent work. Algorithm for sorting an array and algorithm for finding the largest (smallest) element of an array.

Topic 8. *Working with files. Character strings and functions for working with character strings.*

Principles of working with files. The concept of streaming input/output. Binary and text files. Features of storing data in a text file and in a binary file. Functions for working with files. Representing a string as an array of characters. Declaring strings. Standard library functions for working with strings.

Homework assignment. Passing character strings to a function. Functions strcat(), strncat(), strcmp(), strncmp(), strcpy(), strncpy() for working with character strings.

Topic 9. *Structures. Basics of working with structures.*

Structural data types. Syntax for declaring variables of structural data types. Accessing structure elements. Nested structures. Passing a structure as a function argument. Pointers to structures. Accessing a structure field using a pointer to a structure. Arrays of structures. Example of working with structures.

Independent work assignment. Features of storing structures in memory. Union and bit fields. Singly linked and doubly linked lists.

Laboratory lessons

Laboratory work No. 1.

Part 1. Data types and input-output streams

Part 2. Calculation of a mathematical function table.

Laboratory work No. 2.

Part 1. Using the branch operator Part 2. Calculating a definite

integral.

Laboratory work No. 3.

Part 1. Function research

Part 2. Solving nonlinear equations.

Laboratory work No. 4.

Part 1. One-dimensional dynamic arrays.

Part 2. Two-dimensional dynamic arrays.

Laboratory work No. 5.

Topic: Software streams for working with files.

Optional lab assignment that can be completed for extra credit if completed in full:

Lab works #6. Development technologies using the Make compiler GCC for POSIX-compatible operating systems (the topic of the lab work can be changed with the teacher's approval. The lab work is not part of the required assignments. The lab work is done as an extra assignment to get extra points).

The laboratory lessons provide for the consideration of issues that may not be directly included in the list of laboratory work tasks. These issues include the following:

—Number systems (binary, octal, hexadecimal). Direct, reverse, and complementary codes. Single-precision floating-point format (conversion of numbers from the decimal system to binary and from binary to decimal).

— Use of pointers when working with arrays.

— Working with character strings.

— Recursive function calls.

— Features of working with structures.

Verification of the acquisition of relevant knowledge and scoring takes place through the completion of quick tests and modular control work (MCW), which contain tasks covering the specified issues.

Express tests and MODULE TEST are performed during lessons allocated for laboratory work.

6. Independent work of higher education seekers

Independent work involves students preparing for lectures and mastering lecture materials outside of lesson time, as well as mastering specific issues of relevant topics that are assigned for independent study. In addition, independent work must also be carried out as part of preparation for laboratory work, completion of specific items of laboratory work tasks, and preparation of reports. Independent work also includes time for completing individual assignments, such as homework tests, as well as preparing for Module tests and exams. Approximate time allocation for independent work by type of work:

— preparation for lectures, consolidation of lecture materials, independent mastery of individual issues of the relevant topic:

No. Lectures	lesson topic and/or independent work assignments	Number of hours
1	Installation on a PC and introduction to the IDE software interface Code::Blocks. Using the binary number system for storing data in computer and digital technology.	1
2	Binary arithmetic. Algorithms for addition, subtraction, multiplication, division of numbers in the binary number system. IEEE 754 standard for storing single-precision real numbers in computer memory.	1
3	Basic data types: range of values and size in bytes. Bitwise operations. The sizeof operation. The rule for automatic conversion of data types in arithmetic expressions.	2
4	Conditional switch statement. Syntax and structural diagram of the switch statement. Infinite loop. Exiting an infinite loop. Unconditional goto statement. Using the goto statement to performing cyclic actions.	1
5	Type conversion operation applied to pointers. Pointer to pointer. Pointer to function. Passing arguments to the main () function. Recursive function calls.	2

6	Declaration, initialization, and display of multidimensional arrays (using three-dimensional, four-dimensional, and five-dimensional arrays as examples). Determining the number of elements in a static array using the sizeof operation.	1
7	Algorithm for sorting an array and algorithm for finding the largest (smallest) element of an array.	2
8	Passing character strings to a function. Functions strcat(), strncat(), strcmp(), strncmp(), strcpy(), and strncpy() for working with character strings.	3
9	Features of storing structures in memory. Merging and bit fields. Singly linked and doubly linked lists.	2

— Independent work on preparing for laboratory work, completing laboratory tasks, writing reports, and preparing for defense.

No.	Topics of laboratory work	Number of hours
1	Laboratory work No. 1. Part 1. Data types and input/output streams	2
	Part 2. Calculation of a mathematical function table.	3
2	Laboratory work No. 2. Part 1. Using the branch operator	2
	Part 2. Calculating a definite integral.	3
3	Laboratory work No. 3. Part 1. Investigation of the function	2
	Part 2. Solving nonlinear equations.	3
4	Laboratory work No. 4. Part 1. One-dimensional dynamic arrays.	2
	Part 2. Two-dimensional dynamic arrays.	3
5	Laboratory work No. 5. Lr5. Software streams for working with files	3
6	Preparation for the defense of laboratory work and/or completion of an additional task in order to earn extra points for Laboratory work No. 6. Development technologies using the Make compiler GCC for POSIX-compatible operating systems (the topic of the laboratory work may be replaced with another one in agreement with the instructor).	5

— independent work on preparation for the Module test, completion of an individual assignment, and preparation for the exam:

No.	Type of task	Number of hours
1	Preparation for the Module test	5
2	Completing the homework test	12
3	Preparation for the exam	30

Policy and control

7. Policy on academic discipline (educational component)

- ***Recommended teaching methods:*** when studying discipline, it is recommended to use the main and additional literature when mastering the topics of lectures, as well as when

  preparing and performing laboratory work. For successful assimilation of the educational material, it is important to practice the algorithms and solutions discussed in the lectures, as well as those assigned for independent study. It is mandatory to complete laboratory assignments and homework tests, the purpose of which is to acquire practical skills in writing programs and implementing the corresponding algorithms.

- ***lesson attendance rules.*** Attendance at lectures and laboratory work is mandatory. No bonus points are awarded for attendance at lectures or laboratory work is not provided for. Penalty points for absence from lectures or laboratory work are not provided for. During lectures, it is forbidden to distract the teacher from teaching the material with questions that are not related to the topic of the lecture.

- ***Awarding bonus and penalty points.*** Bonus points are awarded for: completing additional laboratory work. Additional points may be awarded for

  the right to perform additional laboratory work when performing research work preparing and publishing a

  scientific article or abstracts of a scientific conference and speaking at a scientific conference, provided that the research work is related to the subject of the discipline. The number of incentive points shall not exceed 10% of the maximum possible rating point that can be obtained for all types of semester assignments submitted for the maximum score.

- ***Academic integrity*** The policy and principles of academic integrity are defined in section 3 Code honor National technical university Ukraine " Igor Sikorsky Kyiv Polytechnic Institute." For more details: https://kpi.ua/code.

- ***Standards of ethical conduct*** Standards of ethical conduct for students and employees are defined in Section 2 of the Code of Honor of the National Technical University of Ukraine "Igor Sikorsky Kyiv Kyiv Polytechnic Institute." For more information, visit: https://kpi.ua/code.

- ***Foreign language instruction*** The academic discipline "Computer Science. Part 1. Fundamentals of Programming and Algorithms" is taught in Ukrainian. In the process of

  teaching the academic discipline, materials in Russian and English may be used.

- ***Informal education.*** Students have the opportunity to gain knowledge on specific topics and sections of the discipline at courses on the Coursera

  (https://www.coursera.org), Prometheus (https://prometheus.org.ua), etc., as blended or additional learning in accordance with the Regulations on the recognition of learning outcomes at Igor Sikorsky Kyiv Polytechnic Institute of acquired in non-formal/informal education (https://osvita.kpi.ua/node/179).

8. Types of control and rating system for assessing learning outcomes (RS)

Types of control:

Ongoing assessment: carried out through questioning during the defense of **laboratory work,** completion of homework assignments (**HCW)**, completion **of Modular control work (MCW)** and **express tests**.

Calendar control: carried out twice per semester to monitor the current status of compliance with the requirements of the discipline's work program (syllabus).

Semester assessment: exam.

Conditions for admission to semester assessment: it is necessary to obtain a minimum sufficient total number of points based on the completion of tasks provided for in the discipline study plan.

The minimum score for admission to the exam is **30 points**.

The maximum number of points for all semester assignments is **70 points**.

The student's rating consists of points earned by completing assignments during the semester as specified in the curriculum, as well as points awarded for passing the exam.
The distribution of points is as follows:
70 points is **the maximum number of points** that can be earned by completing assignments during the semester;
30 points is the **maximum number of points** that can be earned on the exam by completing the **exam ticket** assignments.

Distribution of points for completing assignments during the semester according to the type of assignment:

1. Laboratory work.
Number of main laboratory assignments – 5.
Number of additional laboratory assignments for extra credit – 1
Laboratory works **No. 1...No. 4** consist of two parts — **the first (basic) part** and **the second (main) part**.

The first (basic) part of the lab assignment involves learning the basic concepts of the relevant topic. The maximum number of points awarded for the first (basic) part of the lab assignment is 50% of the maximum number of points awarded for the lab assignment as a whole.

The first (basic) part is mandatory for students who have no experience in programming or for students who have difficulties completing the main part of the laboratory work.

Completion and defense of the first (basic) part of the laboratory work allows the corresponding laboratory work to be credited as completed and defended.

Points awarded for the first (BASIC) part of the laboratory work (applies to laboratory works No. 1...No. 4):

— full mastery of the material (**at least 90%**): **4.50** points.
— good command of the material (**at least 75%**): **3.40. 4.05** points
— sufficient mastery of the material (**at least 60%**): **2.7 3.35** points
— unsatisfactory mastery of the material (**less than 60%)**: **0** points

After completing the first (basic) part of the laboratory work, in order to increase their score, students can complete the tasks of the second (main) part of the laboratory work. In this case, the total maximum number of points for completing the first and second parts of the laboratory work is equal to the maximum number of points awarded for the second (main) part of the laboratory work assignment.

Scoring for completing and defending the MAIN task of the lab work:
— full mastery of the material (at least 90%): **9.00** points.
— good command of the material (at least 75%): **6.75–8.10** points
— sufficient knowledge of the material (at least 60%): **5.40 6.60** points
— unsatisfactory knowledge of the material (less than 60%): **0** points
(***Note***: for laboratory works No. 1. No. 4, the main task of the laboratory work is the second part of the laboratory work).

The maximum number of points for completing laboratory work is 45 points.

Additional points may be awarded for completing additional laboratory work (or preparing a scientific article or abstract for a scientific conference about the discipline).
The maximum number of additional points is 7 points.

2. **Modular control work (MCW).**

Scoring:

— all tasks completed correctly (at least 90%): **8.00** points

— sufficient knowledge of the material (at least 60%), some points of the task are missing or there are errors: **4.80–7.30** points

— insufficient mastery of the material (less than 60%) or no MODULE TEST: **0** points

The maximum number of points for **the Module test** is **8 points**.

3. **Homework assignment.**

The HCT contains 3 tasks.

Task No. 1 and Task No. 2 are mandatory.

Completion of tasks #1 and #2 allows them to be credited in its entirety.

Task No. 3 is **optional** (at the student's discretion) and is completed by students who are aiming for the highest score available for the final coursework.
Scoring:

Task No. 1.

— all task items completed correctly (at least 90%): **3 points**

— sufficient knowledge of the material (at least 60%), some points of the task are missing or there are errors: **1.80–2.50** points

— Insufficient level (less than 60%) or missing task: **0** points
Task No. 2.

— All points of the task completed correctly (at least 90%): **3.00** points

— sufficient mastery of the material (at least 60%), some points of the task are missing or there are errors: **1.80...2.50** points

— Insufficient level (less than 60%) or no assignment: **0** points
Task No. 3.

— all points of the task completed correctly (at least 90%): **3.00** points

— sufficient knowledge of the material (at least 60%), some points of the task are missing or there are errors: **1.80–2.50** points

— Insufficient level (less than 60%) or missing task: **0** points
The maximum number of points for the DCR is **9 points**.

4. **Express tests**.

Small-scale tasks for testing (in writing or in the form of an automated test) to check the level of mastery of the material based on the results of its study in lectures and laboratory lessons.

The maximum number of points for all tests is 8 points.

Test No. 1:
Topic: Number systems. Floating- and fixed-point formats. Scoring for the test:

— Correctly completed tasks: **2.00** points

— Partially correct answer (at least 60%): 1**.20–1.95** points

—Insufficient knowledge of the material (less than 60%) or no answer: — **0** points
Test No. 2:
Topic: Conditional selection operators. Loop operators. Scoring Test:

— Correctly completed tasks: **2.00** points

— Partially correct answer (at least 60%): **1.20...1.95** points

— Insufficient knowledge of the material (less than 60%) or no answer: **0** points

Test No. 3:

Topic: Working with functions. Scoring Test:

— Correctly completed tasks: **2.00** points

— Partially correct answer (at least 60%): **1.20–1.95** points

— Insufficient knowledge of the material (less than 60%) or no answer: **0** points

Test No. 4:

Topic: Working with pointers and arrays. Scoring Test:

— Correctly completed tasks: **2.00** points

— Partially correct answer (at least 60%): **1.20…1.95** points

— Insufficient knowledge of the material (less than 60%) or no answer: **0** points

Students **who scored less than 30 points** must improve their rating by completing the relevant tasks and/or writing an additional test.

Semester assessment: Exam

The exam paper contains three tasks. The points for each task are distributed as follows:

Task 1:

Correct answer: 10 points

Correct solution, with errors (knowledge of the material at least 60%): 6–9 points Insufficient mastery of the material (less than 60%) or no answer: 0 points.

Task 2:

Correct answer: 10 points

Correct procedure given, errors present (knowledge of the material at least 60%): 6–9 points

Insufficient knowledge of the material (less than 60%) or no answer: 0 points

Task 3:

Correct answer: 10 points

Correct execution, existing errors (knowledge of the material not less than 60%): 6–9 points Insufficient knowledge of the material (less than 60%) or no answer: 0 points

In accordance with the Regulations for conducting semester control in remote mode at Igor Sikorsky KPI (https://osvita.kpi.ua/node/148), provided that the higher education applicant has fulfilled the conditions for admission to the semester control event and has scored no less than the passing score for the RS, the conversion of the points earned during the semester to exam points for the discipline is carried out using the formula (rounding the result to the nearest whole number):

R = 60 + 40(Ri – RD)/(Rc – RD)

where: *R* – grade on a 100-point scale;

Ri – the sum of points earned by the student during the semester;

Rc – maximum sum of weighted points for control measures during the semester;

RD – passing score for the exam

Table of correspondence between rating points and university scale grades:

Number of points	*Grade*
100-95	Excellent
94	Very good
84	Good
74-65	Satisfactory
64-60	Sufficient
Less than 60	Unsatisfactory
Admission requirements not met	Not admitted

9. Additional information on the discipline (educational component)

List of topics covered in the module (semester) assessment

- Number systems. Conversion of numbers from the decimal system to binary, hexadecimal, and octal.
- Conversion of numbers from binary, hexadecimal, and octal number systems to the decimal number system.
- Signed and unsigned integers.
- Fixed-point number format. Conversion of real numbers from the decimal system to binary system in fixed-point format.
- Conversion of a real number in the binary system in fixed-point format to the decimal system.
- Direct, reverse, and complementary code. Conversion of negative integers into reverse and complement code.
- Conversion of a number from reverse and complementary code to the decimal number system.
- Arithmetic operations on integers in the binary number system. Going beyond the bit grid. Overflow.
- Number in floating point format. Conversion of a single precision real number to single precision format.
- Converting a single-precision real number in single-precision format to decimal form.
- Performing complex arithmetic operations using the operation precedence table.
- Conditional operators. Using if, if/else, switch, for, while, do/while operators.
- Data types. Explicit conversion of data types. Automatic conversion of data types.
- Functions. Functions that return/do not return values. Passing parameters to a function by value.
- Pointers. Passing variable addresses to a function.
- Static arrays. Declaration, initialization of arrays, assignment of values to array elements.
  Execution operations on arrays : sorting, finding largest/smallest value, median, finding the array element/array element value that occurs most/least frequently in the array.
- Dynamic arrays. Allocating memory for one-dimensional/two-dimensional/three-dimensional dynamic arrays.
- Recursive function call.
- Structured data type. Singly linked and doubly linked lists.

For applicants who have completed distance (online) courses on the subject and completed the corresponding amount of theoretical and practical training, and provided certificate(s) and completed practical assignments of the distance (online) course, assessment on educational component
"Computer Science. Part 1. Fundamentals of Programming and Algorithms" can be determined based on the assessment indicated in the certificate confirming completion of the relevant distance (online) course.

The working program of the academic discipline (syllabus):

Compiled by Senior Lecturer, Ph.D., Serhii Vyshnevyi

Approved by the RTS Department (Minutes No. 06/2024 dated 06/27/2024)

Approved by the Methodological Commission of the Radio Engineering Faculty (Minutes No. 06/2024 dated 28.06.2024)