Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Department of computer technology

# Fundamentals of software development on the Java platform
# Working program of the academic discipline (Syllabus)

## Details of the academic discipline

| | |
|---|---|
| **Level of higher education** | ***First (bachelor's)*** |
| **Branch of knowledge** | *12 Information technologies* |
| **Specialty** | *123 Computer engineering* |
| **Educational program** | *Computer systems and networks* |
| **Discipline status** | *Selective* |
| **Form of education** | *intramural (full-time)/extramural* |
| **Year of training, semester** | *3rd year, autumn semester* |
| **Scope of the discipline** | *4 credit, 120 hours* |
| **Semester control/ control measures** | *Test* |
| **Lessons schedule** | *//rozklad.kpi.ua* |
| **Language of teaching** | *English* |
| **Information about the course leader / teachers** | *Lecturer: senior teacher, Oleksii Aleshchenko*<br>*alexey.aleshchenko@gmail.com*<br>*Laboratory: senior teacher, Oleksii Aleshchenko*<br>*alexey.aleshchenko@gmail.com* |
| **Placement of the course** | *//comsys.kpi.ua* |

## Program of educational discipline

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

*The discipline "Fundamentals of software development on the Java platform " is aimed at students mastering the methods and means of programming tasks of arbitrary complexity using the methodology based on the object-oriented paradigm, using the Java platform. Knowledge of the basics of software development on the Java platform is necessary for creating software for computer systems, real-time systems, Internet applications, and mobile devices.*

*The purpose of teaching the discipline "Fundamentals of software development on the Java platform " is:*

***ABILITY*** *: for abstract thinking, analysis and synthesis; work in a team; use modern methods and programming languages to develop algorithmic and software; create system and application software for computer systems and networks; to ensure the protection of information processed in computer and cyber -physical systems and networks for the purpose of implementing the established information security policy; systematically administer, use, adapt and operate existing information technologies and systems; identify, classify and describe the operation of software and technical means, computer and cyber -physical systems, networks and their components by using analytical methods and modeling methods; design, implement, administer and maintain global, local intelligent software- configured computer networks; develop, adapt, use software to improve the efficiency of high-performance computer systems;*

*organization of computing processes in high-performance computer systems with different structural organization based on the use of the latest planning and dispatching technologies and modern operating systems;*

***KNOWLEDGE:*** *basics of project management; to identify, formulate and solve technical problems of the specialty, using the methods that are most suitable for achieving the set goals ; technical characteristics, design features, purpose and rules of operation of software and technical means of computer systems and networks for solving technical problems of the specialty;*

***SKILLS*** *: conducting experiments, collecting data and modeling in computer systems; apply knowledge to identify, formulate and solve technical problems of the specialty, using methods that are most suitable for achieving the set goals; to solve problems of analysis and synthesis of means characteristic of the specialty; apply knowledge of technical characteristics, design features, purpose and rules of operation of software and technical means of computer systems and networks to solve technical problems of the specialty; develop software for embedded and distributed applications, mobile and hybrid systems, calculate, operate equipment typical for the specialty; perform experimental research on professional topics; perform parameter calculations of individual computer systems, computer networks; be able to create and maintain databases.*

## 2. 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. Part 1. Programming", "Data structures and algorithms", "Higher mathematics. Part 1", "Analytic geometry and linear algebra".*

## 3. Content of the academic discipline

*Topic 1. General characteristics of the Java language. Comparison with other programming languages. Simple Types and Basics of Java Syntax.*

*Topic 2. Classes and methods in Java. Packages and interfaces.*

*Topic 3. Working with arrays in Java.*

*Topic 4. Working with deadlines in Java. String classes, String Buffer, String Builder.*

*Topic 5. Method overloading. Constructor overloading. Access control. Static elements.*

*Topic 6. Properties (fields). Inheritance of properties. Inner and anonymous classes.*

*Topic 7. Creation of multi-level hierarchies. Object class.*

*Topic 8. Methods. Imitation of methods. Polymorphism. Super and final specifiers.*

*Topic 9. Implementation of polymorphism. Overriding methods. Early and late binding.*

*Topic 10. Collections in Java. Linked lists.*

*Topic 11. Exception handling and unit testing.*

*Topic 12. Memory management, object destructors and Garbage Collector.*

*Topic 13. Package java.util.stream.*

*Topic 14. Events and event-based management of application behavior.*

## 4. Educational materials and resources

*Basic :*

1. *Java programming: computer workshop [Electronic resource]: teaching. help _ for studies _ specialty 122 "Computer science", educational and professional program "Computer monitoring and geometric modeling of processes and systems" / KPI named after Igor Sikorskyi; editor: Yu. A. Tarnavskyi. – Electronic text data (1 file: 686 Kbytes ). – Kyiv: KPI named after Igor Sikorskyi, 2021. - 95 p. https://ela.kpi.ua/bitstream/123456789/41885/1/Java-programming.pdf*
2. *Synopsis of lectures from the software module "Object-oriented programming" / Developer: assistant Oleksiy Vadimovich Aleshchenko. Department of Computer Engineering FIOT. Approved at the department meeting (Minutes No. 11 dated May 24, 2017)*

https://comsys.kpi.ua/katalog/files/konspekt-lekciy-oop-1.pdf

3. Fundamentals of software development on the Java platform. Methodical instructions for performing laboratory work for bachelor's degree holders in the majors: 121 Software engineering, 123 Computer engineering, 126 Information systems and technologies of all forms of education / Compiled by: Aleschenko O.V. - K.: KPI named after Igor Sikorskyi, 2022. [Electronic resource]. Approved by the Department of Computing (Protocol No. 10 dated 05/25/2022). Agreed by the Methodical Commission of FIOT (protocol No. 10 dated 09.06.2022). https://comsys.kpi.ua/methodichni-vkazannya-po-disciplinam.

Additional :

4. Tsybulnyk S. O. Software development technologies-1. Computer workshop [Electronic resource]: teaching. help _ for studies _ specialty 151 "Automation and computer-integrated technologies", educational and professional program "Computer-integrated technologies and navigation and control systems" / S. O. Tsybulnyk ; KPI named after Igor Sikorsky. Kyiv: KPI named after Igor Sikorskyi, 2021. 126 p. https://ela.kpi.ua/bitstream/123456789/41516/3/Tekhnolohii-rozrobky-prohramnoho-zabezpechennia_KompPrakt.pdf

## Educational content

### 5. Methods of mastering an educational discipline (educational component)

**Full-time education**

| Names of sections and topics | Number of hours | | | | |
|---|---|---|---|---|---|
| | In total | including | | | |
| | | Lectures | Practical (seminar) | Laboratory work | SRS |
| Topic 1. General characteristics of the Java language. Comparison with other programming languages. Simple Types and Basics of Java Syntax. | 9 | 2 | - | 3 | 4 |
| Topic 2. Classes and methods in Java. Packages and interfaces. | 6 | 2 | - | - | 4 |
| Topic 3. Working with arrays in Java. | 9 | 2 | - | 3 | 4 |
| Topic 4. Working with deadlines in Java. String classes, String Buffer, String Builder. | 9 | 2 | - | 3 | 4 |
| Topic 5. Method overloading. Constructor overloading. Access control. Static elements. | 6 | 2 | - | - | 4 |
| Topic 6. Properties (fields). Inheritance of properties. Inner and anonymous classes. | 11 | 4 | - | 3 | 4 |
| Topic 7. Creation of multi-level | 9 | 2 | - | 3 | 4 |

|  |  |  |  |  |  |
|---|---|---|---|---|---|
| *hierarchies. Object class.* |  |  |  |  |  |
| *Topic 8. Methods. Imitation of methods. Polymorphism. Super and final specifiers.* | 9 | 2 | - | 3 | 4 |
| *Topic 9. Implementation of polymorphism. Overriding methods. Early and late binding.* | 6 | 2 | - | - | 4 |
| *Topic 10. Collections in Java. Linked lists.* | 10 | 6 | - | - | 4 |
| *Topic 11. Exception handling and unit testing.* | 8 | 4 | - | - | 4 |
| *Topic 12. Memory management, object destructors and Garbage Collector.* | 6 | 2 | - | - | 4 |
| *Topic 13. Package java.util.stream.* | 7 | 2 | - | - | 5 |
| *Topic 14. Events and event-based management of application behavior.* | 7 | 2 | - | - | 5 |
| ***MCW*** | 4 | - | - | - | 4 |
| ***Test*** | 4 | - | - | - | 4 |
| **Total in the semester:** | 120 | 36 | - | 18 | 66 |

**Correspondence form teaching**

| *Names of sections and topics* | *Number of hours* |  |  |  |  |
|---|---|---|---|---|---|
|  | *In total* | *including* |  |  |  |
|  |  | *Lectures* | *Practical (seminar)* | *Laboratory work* | *SRS* |
| *Topic 1. General characteristics of the Java language. Comparison with other programming languages. Simple Types and Basics of Java Syntax.* | 9 | 1 | - | 2 | 6 |
| *Topic 2. Classes and methods in Java. Packages and interfaces.* | 7 | 1 | - | - | 6 |
| *Topic 3. Working with arrays in Java.* | 7,25 | 0,25 | - | 1 | 6 |
| *Topic 4. Working with deadlines in Java. String classes, String Buffer, String Builder.* | 7,25 | 0,25 | - | 1 | 6 |
| *Topic 5. Method overloading. Constructor overloading. Access control. Static elements.* | 6,25 | 0,25 | - | - | 6 |

| | | | | | |
|---|---|---|---|---|---|
| *Topic 6. Properties (fields). Inheritance of properties. Inner and anonymous classes.* | 7,25 | 0,25 | - | 1 | 6 |
| *Topic 7. Creation of multi-level hierarchies. Object class.* | 9 | 1 | - | 2 | 6 |
| *Topic 8. Methods. Imitation of methods. Polymorphism. Super and final specifiers.* | 7,25 | 0,25 | - | 1 | 6 |
| *Topic 9. Implementation of polymorphism. Overriding methods. Early and late binding.* | 7,25 | 0,25 | - | - | 7 |
| *Topic 10. Collections in Java. Linked lists.* | 8 | 1 | - | - | 7 |
| *Topic 11. Exception handling and unit testing.* | 7,25 | 0,25 | - | - | 7 |
| *Topic 12. Memory management, object destructors and Garbage Collector.* | 7,25 | 0,25 | - | - | 7 |
| *Topic 13. Package java.util.stream.* | 8 | 1 | - | - | 7 |
| *Topic 14. Events and event-based management of application behavior.* | 8 | 1 | - | - | 7 |
| ***MKR*** | 7 | - | - | - | 7 |
| ***Test*** | 7 | - | - | - | 7 |
| **Total in the semester:** | 120 | 8 | - | 8 | 104 |

***Lecture classes***

***Full-time education***

| *lecture no* | *The title of the lecture topic, the list of main questions and tasks for the SRS* |
|---|---|
| *1* | ***"Introduction to Java ".*** *General characteristics of the Java language. Comparison with other programming languages. Simple Types and Basics of Java Syntax.*<br>***SRS:*** *Partially or completely complete the first laboratory work of this credit module in the Java programming language.* |
| *2* | ***"Introduction to Java. Continuation".*** *Classes and methods in Java. Packages and interfaces.*<br>***CRC:*** *Create multiple methods in multiple different classes that reside in different packages and implement different interfaces.* |
| *3* | ***"Working with arrays in Java ".*** *One-dimensional, multi-dimensional arrays and objects representing arrays in Java.*<br>***SRS:*** *Partially or completely complete the second laboratory work of this credit module in the Java programming language.* |
| *4* | ***"Working with deadlines in Java ".*** *String classes, String Buffer, String Builder.* |

| | |
|---|---|
| | ***SRS:*** *Partially or completely complete the third laboratory work of this credit module in the Java programming language.* |
| *5* | ***"Class methods. Overloading, static and access".*** *Method overloading. Constructor overloading. Access control. Static elements.*<br>***CRC:*** *Create multiple overloaded methods with different access modifiers.* |
| *6* | ***"Properties (fields) of the class. Anonymous classes on the example of sorting an array of objects".*** *Anonymous classes. Comparator interface. Compare method. Sorting an array of objects. Arrays class. The Sort method.*<br>***SRS:*** *Partially or completely complete the fourth laboratory work of this credit module in the Java programming language.* |
| *7* | ***" Imitation and Visibility Management".*** *Properties (fields). Inheritance of properties. Visibility management.*<br>***CRC:*** *Create a class with multiple fields with different scopes. Create a class that inherits from the previous one. Try to access the fields of the previous class in it and create your own fields with the same names.* |
| *8* | ***"Hierarchies of objects".*** *Creation of multi-level object hierarchies through inheritance and/or aggregation. Object class.*<br>***SRS:*** *Partially or fully complete the fifth laboratory work of this credit module in the Java programming language.* |
| *9* | ***"Imitation of methods. Polymorphism".*** *Imitation of methods. Polymorphism. Super and final specifiers.*<br>***SRS:*** *Partially or fully complete the sixth laboratory work of this credit module in the language Java programming.* |
| *10* | ***"Implementation of polymorphism".*** *Implementation of polymorphism. Overriding methods. Early and late binding.*<br>***CRC:*** *Create a class with several methods. Create a class that inherits from the previous one and overrides its methods. Create an instance of the second class, the reference to which will store the variable of the first class. Try to access through the specified variable of the first class to the overridden method from the second class.* |
| *11* | ***"Collections in Java ".*** *Collections in Java. Linked lists. Set and List interfaces.*<br>***SRS:*** *Try to create own class which _ implements any _ interface collections (without implementing an iterator and equivalence).* |
| *12* | ***" Collections in Java. Continuation ".*** *toArray methods. Iterators.*<br>***SRS:*** *Implement an iterator within the previous SRS.* |
| *13* | ***" Collections in Java. Continuation ".*** *Uniqueness of elements in collections of the Set type. Equivalence of objects.*<br>***SRS:*** *Implement the equivalence check of individual elements and the entire collection within the framework of the previous SRS.* |
| *14* | ***"Exception Handling".*** *Classes of description of exceptional situations. Own exceptional situations. Exception, Error, Runtime Exception.*<br>***SRS:*** *Create your own class to represent and handle your own exception.* |
| *15* | ***"Modular testing".*** *Building and using unit tests.*<br>***SRS:*** *Create your own class with methods for unit testing one of the classes written as part of one of the laboratory works of this credit module.* |
| *16* | ***"Memory Management, Object Destructors and Garbage Collector ".*** *The "garbage collection" system, the criterion for the presence of references to objects, the destructor.*<br>***SRS:*** *Familiarize yourself with the algorithms of the "garbage collection" system ( Garbage Collector ) in Java.* |

| | |
|---|---|
| 17 | **" Package java.util.stream ".** Functional programming style. Pure functions. Map-reduce technology.<br>**SRS:** Implement the algorithm of selection from an array by criteria using the java.util.stream package. |
| 18 | **"Events and Event-Oriented Control of Program Behavior".** Events in Java. AWT and Swing modules. Observer, JMS.<br>**CSC:** Implement a program in the Java programming language that uses the event mechanism. |

### External form of education

| lecture no | The title of the lecture topic, the list of main questions and tasks for the SRS |
|---|---|
| 1 | **"Introduction to Java ".** General characteristics of the Java language. Comparison with other programming languages. Simple Types and Basics of Java Syntax.<br>**SRS:** Partially or completely complete the first laboratory work of this credit module in the Java programming language. |
| | **"Introduction to Java. Continuation".** Classes and methods in Java. Packages and interfaces.<br>**CRC:** Create multiple methods in multiple different classes that reside in different packages and implement different interfaces. |
| 2 | **"Working with arrays in Java ".** One-dimensional, multi-dimensional arrays and objects representing arrays in Java.<br>**SRS:** Partially or completely complete the second laboratory work of this credit module in the Java programming language. |
| | **"Working with deadlines in Java ".** String classes, String Buffer, String Builder.<br>**SRS:** Partially or completely complete the third laboratory work of this credit module in the Java programming language. |
| | **"Class methods. Overloading, static and access".** Method overloading. Constructor overloading. Access control. Static elements.<br>**CRC:** Create multiple overloaded methods with different access modifiers. |
| | **"Properties (fields) of the class. Anonymous classes on the example of sorting an array of objects. Imitation and Visibility Management".** Properties (fields). Inheritance of properties. Visibility management. Anonymous classes. Comparator interface. Compare method. Sorting an array of objects. Arrays class. The Sort method.<br>**CRC:** Create a class with multiple fields with different scopes. Create a class that inherits from the previous one. Try to access the fields of the previous class in it and create your own fields with the same names. Partially or completely complete the fourth laboratory work of this credit module in the Java programming language. |
| | **"Hierarchies of objects".** Creation of multi-level object hierarchies through inheritance and/or aggregation. Object class.<br>**SRS:** Partially or fully complete the fifth laboratory work of this credit module in the Java programming language. |
| 3 | **"Imitation of methods. Polymorphism".** Imitation of methods. Polymorphism. Super and final specifiers.<br>**SRS:** Partially or fully complete the sixth laboratory work of this credit module in the language Java programming. |
| | **"Implementation of polymorphism".** Implementation of polymorphism. Overriding methods. Early and late binding. |

| | |
|---|---|
| | ***CRC:*** *Create a class with several methods. Create a class that inherits from the previous one and overrides its methods. Create an instance of the second class, the reference to which will store the variable of the first class. Try to access through the specified variable of the first class to the overridden method from the second class.* |
| | ***"Collections in Java ".*** *Collections in Java. Linked lists. Set and List interfaces. Collections in Java. Linked lists. Set and List interfaces. toArray methods. Iterators. Uniqueness of elements in collections of the Set type. Equivalence of objects.*<br>***SRC:*** *Try to create a custom class that implements any collection interface with an iterator and equivalence implementation.* |
| | ***"Exception Handling and Unit Testing".*** *Classes of description of exceptional situations. Own exceptional situations. Exception, Error, Runtime Exception. Building and using unit tests.*<br>***SRS:*** *Create your own class to represent and handle your own exception. Create your own class with methods for unit testing one of the classes written as part of one of the laboratory works of this credit module.* |
| | ***"Memory Management, Object Destructors and Garbage Collector ".*** *The "garbage collection" system, the criterion for the presence of references to objects, the destructor.*<br>***SRS:*** *Familiarize yourself with the algorithms of the "garbage collection" system ( Garbage Collector ) in Java.* |
| *4* | ***"Package java.util.stream".*** *Functional programming style. Pure functions. Map-reduce technology.*<br>***SRS:*** *Implement the algorithm of selection from an array by criteria using the java.util.stream package.* |
| | ***"Events and Event-Oriented Control of Program Behavior".*** *Events in Java. AWT and Swing modules. Observer, JMS.*<br>***CSC:*** *Implement a program in the Java programming language that uses the event mechanism.* |

### *Laboratory classes*

***Full-time education***

| *No. z/p* | *The name of the laboratory work* | *Number of aud. hours* |
|---|---|---|
| *1* | *The main types and operators of the Java programming language (Topic 1)* | *3* |
| *2* | *Arrays that are represented as objects (Topic 3)* | *3* |
| *3* | *Strings that are represented as objects (Topic 4)* | *3* |
| *4* | *Classes and their properties (Topic 6)* | *3* |
| *5* | *Relations between classes, construction of hierarchies (Topic 7)* | *3* |
| *6* | *Inheritance and polymorphism (Topic 8)* | *3* |
| | ***Together:*** | *18* |

***External form of education***

| *No. z/p* | *The name of the laboratory work* | *Number of aud. hours* |
|---|---|---|

| | | |
|---|---|---|
| 1 | The main types and operators of the Java programming language (Topic 1) | 2 |
| 2 | Arrays that are represented as objects (Topic 3) | 1 |
| 3 | Strings that are represented as objects (Topic 4) | 1 |
| 4 | Classes and their properties (Topic 6) | 1 |
| 5 | Relations between classes, construction of hierarchies (Topic 7) | 2 |
| 6 | Inheritance and polymorphism (Topic 8) | 1 |
| | **Together:** | 8 |

## 6. Independent work of student

***Full-time education***

| No. z/p | The name of the topic submitted for independent processing | Number of hours of SRS |
|---|---|---|
| 1 | **"Introduction to Java ".** Partially or completely complete the first laboratory work of this credit module in the Java programming language. | 4 |
| 2 | **"Introduction to Java. Continuation".** Create several methods in several different classes that are in different packages and implement different interfaces. | 4 |
| 3 | **"Working with arrays in Java ".** Partially or completely complete the second laboratory work of this credit module in the Java programming language. | 4 |
| 4 | **"Working with deadlines in Java ".** Partially or completely complete the third laboratory work of this credit module in the Java programming language. | 4 |
| 5 | **"Class methods. Overloading, static and access".** Create multiple overloaded methods with different access modifiers. | 4 |
| 6 | **"Properties (fields) of the class. Anonymous classes on the example of sorting an array of objects".** Partially or completely complete the fourth laboratory work of this credit module in the Java programming language. | 2 |
| 7 | **" Imitation and Visibility Management".** Create a class with multiple fields with different scopes. Create a class that inherits from the previous one. Try to access the fields of the previous class in it and create your own fields with the same names. | 2 |
| 8 | **"Hierarchies of objects".** Partially or completely complete the fifth laboratory work of this credit module in the Java programming language. | 4 |
| 9 | **"Imitation of methods. Polymorphism".** Partially or completely complete the sixth laboratory work of this credit module in the language Java programming. | 4 |
| 10 | **"Implementation of polymorphism".** Create a class with several methods. Create a class that inherits from the previous one and overrides its methods. Create an instance of the second class, the reference to which will store the variable of the first class. Try to access through the specified variable of the first class to the overridden method from the second class. | 4 |
| 11 | **"Collections in Java ".** Try to create a custom class that implements any collection interface (without implementing an iterator and equivalence). | 2 |
| 12 | **" Collections in Java. Continuation ".** Implement an iterator within the framework of the previous SRS. | 1 |

| | | |
|---|---|---|
| *13* | ***" Collections in Java. Continuation ".*** *Implement the verification of the equivalence of individual elements and the entire collection within the framework of the previous SRS.* | *1* |
| *14* | ***"Exception Handling".*** *Create your own class to represent and handle your own exception.* | *2* |
| *15* | ***"Modular testing".*** *Create your own class with methods for unit testing one of the classes written as part of one of the laboratory works of this credit module.* | *2* |
| *16* | ***"Memory Management, Object Destructors and Garbage Collector ".*** *Familiarize yourself with the algorithms of the "garbage collection" system ( Garbage Collector ) in Java.* | *4* |
| *17* | ***" Package java.util.stream ".*** *Implement the algorithm of selection from the array according to the criterion using the java.util.stream package.* | *5* |
| *18* | ***"Events and Event-Oriented Control of Program Behavior".*** *Implement a program in the Java programming language that uses the event mechanism.* | *5* |
| *19* | ***"Preparation for Modular control work (MCW)"*** | *4* |
| *20* | ***"Preparation for the test"*** | *4* |
| | ***Together:*** | *66* |

***External form of education***

| *No. z/p* | *The name of the topic submitted for independent processing* | *Number of hours of SRS* |
|---|---|---|
| *1* | ***"Introduction to Java ".*** *Partially or completely complete the first laboratory work of this credit module in the Java programming language. Create several methods in several different classes that are in different packages and implement different interfaces.* | *12* |
| *2* | ***"Working with arrays in Java ".*** *Partially or completely complete the second laboratory work of this credit module in the Java programming language.* | *6* |
| *3* | ***"Working with deadlines in Java ".*** *Partially or completely complete the third laboratory work of this credit module in the Java programming language.* | *6* |
| *4* | ***"Class methods. Overloading, static and access".*** *Create multiple overloaded methods with different access modifiers.* | *6* |
| *5* | ***"Properties (fields) of the class. Anonymous classes on the example of sorting an array of objects. Imitation and Visibility Management".*** *Create a class with multiple fields with different scopes. Create a class that inherits from the previous one. Try to access the fields of the previous class in it and create your own fields with the same names. Partially or completely complete the fourth laboratory work of this credit module in the Java programming language.* | *6* |
| *6* | ***"Hierarchies of objects".*** *Partially or completely complete the fifth laboratory work of this credit module in the Java programming language.* | *6* |
| *7* | ***"Imitation of methods. Polymorphism".*** *Partially or completely complete the sixth laboratory work of this credit module in the language Java programming.* | *6* |
| *8* | ***"Implementation of polymorphism".*** *Create a class with several methods. Create a class that inherits from the previous one and overrides its methods. Create an instance of the second class, the reference to which will store the variable of the first class. Try to access through the specified variable of the first class to the overridden method from the second class.* | *7* |

| | | |
|---|---|---|
| *9* | ***"Collections in Java ".*** *Try creating a custom class that implements any collection interface with an iterator and equivalence implementation.* | *7* |
| *10* | ***"Exception Handling and Unit Testing".*** *Create your own class to represent and handle your own exception. Create your own class with methods for unit testing one of the classes written as part of one of the laboratory works of this credit module.* | *7* |
| *11* | ***"Memory Management, Object Destructors and Garbage Collector ".*** *Familiarize yourself with the algorithms of system operation and "garbage collection". Collector) in Java.* | *7* |
| *12* | ***" Package java.util.stream ".*** *Implement the algorithm of selection from the array according to the criterion using the java.util.stream package.* | *7* |
| *13* | ***"Events and Event-Oriented Control of Program Behavior".*** *Implement a program in the Java programming language that uses the event mechanism.* | *7* |
| *14* | ***"Preparation for Modular control work (MCW)"*** | *7* |
| *15* | ***"Preparation for the test"*** | *7* |
| | ***Together:*** | *104* |

## Policy and control

### 7. Policy of academic discipline (educational component)

- rules for the protection of laboratory work:
  - the student provides the program code for laboratory work,
  - the student defends the provided software code through an interview;
- deadline and rescheduling policy:
  - the student has the right to submit and resubmit laboratory work up to and including the day of assessment,
  - there is no limit on the number of folds;
- academic integrity policy:
  - the student has the right to protect the program code that was written not only by him, but in this case the student must understand "what" and "how" this code performs, be able to use it as a basis for making modifications that may be suggested by the teacher.

### 8. Types of control and rating system for evaluating learning outcomes (RSO)

*Types of control from the educational discipline "Fundamentals of software development on the Java platform " include:*

***Laboratory works:***

*Independent performance of six laboratory works is planned. The topics of laboratory works are coordinated in time and content with the topics of lectures. Carrying out laboratory work in full allows you to acquire practical programming skills on the Java platform.*

***Current control:***

*It is planned to carry out control work (MKR)*

***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*

*The student's semester rating consists of the points he receives for the types of work in accordance with Table 1.*

*Table 1*

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

| ***Type of educational work*** | ***Total by type of work*** |
|---|---|
| *Performance and protection of laboratory work No. 1* | *15* |
| *Performance and protection of laboratory work No. 2* | *15* |
| *Performance and protection of laboratory work No. 3* | *15* |
| *Performance and protection of laboratory work No. 4* | *15* |
| *Performance and protection of laboratory work No. 5* | *15* |
| *Performance and protection of laboratory work No. 6* | *15* |
| *MCW* ***Rk*** | *10* |
| ***Rp*** | ***100*** |
| ***( Rz ) as desired*** | ***20*** |

*The individual current rating of the student (* ***Rp*** *) consists of the points he receives for performing laboratory work (* ***Rl )*** *and MKR* ***( Rk*** *). During the semester, students perform 6 laboratory works. The maximum number of points for each laboratory work is 15. Points are awarded for:*

- *theoretical component – 7 points,*
- *practical component - 8 points.*

*The maximum possible score for laboratory work is 15 points. The maximum number of points for all laboratory works is 6 x 15 = 90 points.*

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

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

***Rp = Rl + Rk,*** *where Rp is the student's semester rating (MCR, laboratory work).*

*A necessary condition for a student's admission to credit is his individual semester rating ( Rp ), not less than 40 points, and the absence of arrears from laboratory work and MKR. If the mentioned requirements are not met, the student will not be admitted to the credit.*

*Table of correspondence of rating points to grades on the university scale:*

| ***Scores*** | ***Rating*** |
|---|---|
| *100-95* | *Perfectly* |
| *94-85* | *Very good* |
| *84-75* | *Fine* |
| *74-65* | *Satisfactorily* |

| | |
|---|---|
| *64-60* | *Enough* |
| *Less than 60* | *Unsatisfactorily* |
| *Admission conditions not met* | *Not allowed* |

## 9. Additional information on the discipline (educational component)

*As part of the study of the discipline "Fundamentals of software development on the Java platform ", crediting of points obtained as a result of distance courses is allowed, provided that the program of this course is agreed with the teacher.*

*Working program of the academic discipline (syllabus):*<br>
***Compiled by** Oleksii Aleshchenko, a senior lecturer at the Department of Computing*<br>
***Approved** by the Department of Computing (Protocol No. 10 dated 25 May 2022)*<br>
***Agreed** by the Methodical Commission of the faculty (protocol No. 10 dated 09 June 2022)*