МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

УТВЕРЖДАЮ

Заведующий кафедрой
Программирования и информационных технологий
_____ *проф. Махортов С.Д,*
03.05.2023

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

Б1.О.35 Введение в программирование

1. Код и наименование направления подготовки/специальности:

10.03.01 Информационная безопасность

2. Профиль подготовки/специализация:

Безопасность компьютерных систем

3. Квалификация (степень) выпускника: Бакалавр

4. Форма обучения: Очная

5. Кафедра, отвечающая за реализацию дисциплины:

Программирования и информационных технологий

6. Составители программы:

ст. преподаватель каф. ПиИТ Соломатин Дмитрий Иванович
e-mail: solomatin@cs.vsu.ru
факультет: Компьютерных наук
кафедра: Программирования и информационных технологий

7. Рекомендована:

НМС ф-та компьютерных наук, протокол № 7 от 03.05.2023

_____
_____
_____

8. Учебный год: 2023-2024 **Семестр(ы):** 1

9.Цели и задачи учебной дисциплины:

Изучение студентами основ программирования и принципов проектирования программ, а также овладение практическими навыками написания относительно простых программ (на конкретном языке).

10. Место учебной дисциплины в структуре ООП:

Учебная дисциплина относится к обязательной части блока Б1.

Для успешного освоения дисциплины необходимы знания математики и основ информатики в объеме школьной программы.

11. Планируемые результаты обучения по дисциплине/модулю (знания, умения, навыки), соотнесенные с планируемыми результатами освоения образовательной программы (компетенциями выпускников):

Код	Название компетенции	Код(ы)	Индикатор(ы)	Планируемые результаты обучения
ОПК-7	Способен использовать языки программирования и технологии разработки программных средств для решения задач профессиональной деятельности.	ОПК-7.1	Знает основные принципы построения компьютера, формы и способы представления данных в персональном компьютере.	Знать: Основные принципа построения компьютера, формы и способы представления данных в программе.
		ОПК-7.2	Знает области и особенности применения языков программирования высокого уровня.	Знать: Области и особенности применения языков программирования высокого уровня.
		ОПК-7.3	Знает язык программирования высокого уровня (структурное, объектно-ориентированное программирование).	Знать: Язык программирования высокого уровня Java, структурное и объектно-ориентированное программирование.
		ОПК-7.4	Умеет работать с интегрированной средой разработки программного обеспечения	Уметь: Работать в интегрированной среде разработки программного обеспечения Java (Intelliji Idea).
		ОПК-7.5	Умеет разрабатывать и реализовывать на языке высокого уровня алгоритмы решения типовых профессиональных задач	Уметь: Разрабатывать и реализовывать алгоритмы решения задач на языке высокого уровня Java.
		ОПК-7.6	Владеет навыками разработки, документирования, тестирования и отладки программ	Владеть: Навыками разработки, документирования, тестирования и отладки программ на языке Java.

12. Объем дисциплины в зачетных единицах/час. (в соответствии с уч. планом) **–** 4 / 144.

Форма промежуточной аттестации – Экзамен

13. Виды учебной работы

Вид учебной работы		Трудоемкость			
		Всего	По семестрам		
			1 сем.	–	–
Аудиторные занятия		66	66	–	–
в том числе:	лекции	34	34	–	–

	практические	16	16	–	–
	лабораторные	16	16	–	–
Самостоятельная работа		42	42	–	–
в том числе: курсовая работа (проект)		–	–	–	–
Форма промежуточной аттестации (зачет – 0 час. / экзамен – 36 час.)		36	36	–	–
Итого:		144	144	–	–

13.1. Содержание дисциплины

№ п/п	Наименование раздела дисциплины	Содержание раздела дисциплины	Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК
1. Лекции			
1.1	Введение в предмет	Цели и задачи изучения дисциплины; понятие и свойства алгоритма; краткий обзор языка Java; примеры и разбор простейших программ на языке Java; краткий обзор сред разработки Java-программ; создание проекта в среде разработки	
1.2	Переменные и типы данных, ввод-вывод данных	Понятие переменной, понятие типа данных и строгой типизации; стандартные типы языка Java (примитивные и String), преобразования типов; особенности хранения различных типов в памяти компьютера и их обработки; System.in и System.out, ввод данных с помощью класс java.util.Scanner, форматирование вывода с помощью printf и String.format	
1.3	Функции	Понятие функций (статических методов класса применительно к Java): описание и вызов, передача параметров; проектирование программы с использованием функций; важность разделения программы на подзадачи и превильного выделения подзадач, структуризация и принципы повторного использования кода; примеры программ с использованием функций	
1.4	Операторы управления ходом выполнения программ	Условный оператор, составной оператор, тернарный оператор, операторы циклов в Java, операторы break и continue; вложенные блоки кода и видимость пеменных; соглашения по оформлению Java-кода; примеры решения задач	
1.5	Простейшие алгоритмы	Примеры решения задач: перевод десятичного числа в двоичное представление, собственная реализация sqrt методом половинного деления, вычисление числа Пи, выделение n-ой части строки, разделенной запятыми, печать n символов последовательности и др.; разные варианты решения задач и выбор оптимального варианта	
1.6	Составные типы данных	Массивы и множественные массивы (массивы массивов); типичные задачи обработки массивов, класс java.util.Arrays; разработка библиотеки функций ввода/вывода массивов и других функции в виде класс ArrayUtils. Перечисления (enum). Структыры данных в виде простейших классов, важность применения структур для упрощения и струкутризация кода, массивы структур, примеры задач. Массивы и структуры в качестве параметров функций и возвращаемых значений. Понятие типов-значений (value types) и ссылочных типов (reference types) данных в Java, понятие объектов, ссылочная модель и сборка мусора.	

1.7	Строки	Строки в Java, особенности класса String - особенности реализации (неизменяемость), поддерживаемые методы, сравнение строк с помощью equals. Особенности конкатенации строк и класс StringBuilder. Поверхностное знакомство с регулярными выражения и возможностями их применения (RegExp и некоторые методы String).	
1.8	Типичные задачи обработки массивов и строк	Примеры решения задач: поиск минимума/максимума в массиве, поиск индекса элемента, сортировка массива методом "пузырька", передача различных критериев сортировки в метод Arrays.sort, бинарный поиск в упорябоченном массиве, операции со столбцами и строка в двумерном массиве и др.; демострации важности структуризации программы с помощью фунций на примере задач обработки массивов и строк	
1.9	Основы объектно-ориентированного подхода	Принципы объектно ориентированного подхода; понятие класса и его экземпляров; различия между static-функциями и методами классов; классы как расширение концепции типа данных в виде объединения данных и методов их обработки, понятие состояния объекта, примеры; принципы инкапсуляции; понятие наследования и полиморфизма; класс Object и его методы	
1.10	Создание приложений с оконным интерфейс	Принципы создания приложений с оконным интерфейсом; библиотека Swing и базовые Swing-компоненты; визуальное проектирование форм (JFrame) в среде разработки, обработка событий компонентов; понятие модели данных для сложных копонентов, JTable и разработка класса JTableUtils для упрощения работы с JTable. Типичная архитектура приложений с оконным интерфейсом и важность разделения логики и отображения, приложения с несколькими формами, примеры приложений. Второй вариант построения оконного интерфейса - с помощью JavaFX, возможности и особенности, примеры приложений.	
1.11	Коллекции	Понятие коллекций, какие виды колдекций бывают (списки, словари, множества, стеки и очереди), иерархия коллекций в Java; списки (List<T>) по сравнению с массивами, методы списков; словари (Map<K, V>) и множества (Set<T>) и их методы; понятие обобщенных типов данных и кода (generics); разница между интерфейсом и реализацией (List<T> и ArrayList<T>/LinkedList<T>, Map<K, V> и TreeMap<K, V>/HashMap<K, V>); примеры эффективного решения задач с помощью коллекций	
1.12	Создание прикладных приложений (создание игры)	Пример создания полноценного оконного приложения на Java - игры "Сапер" в качестве демонстрации применения структур данных (классов, массивов) и алгоритмов применительно к конкретной практической задаче; архитектура приложения с применение ООП-подхода - выделение логики в виде отдельно класса, реализация отображение в JTable; знакомство с классом java.awt.Graphics	

1.13	Рекурсия и рекурсивные алгоритмы	Понятие рекурсии в программирование; прямая и косвенная рекурсия; вычисление факториала, когда стоит и когда не стоит использовать рекурсию; рекурсивное вычисление чисел Фибоначчи - экспоненциальный рост количества повторных вызовов функций в некоторых рекурсивных алгоритмах и возможное решения с помощью кеширования результатов вычислений; примеры решения задач: рисование треугольника Серпинского, сопоставление строки шаблону, обход двумерного поля в глубину и ширину; варианты реализации истинно рекурсивных алгоритмов без рекурсивных вызовов с применение стеков (Stack<T>) и очередей (Queue<T>)	
2. Практические занятия			
2.1	Переменные и типы данных, ввод-вывод данных	Понятие переменной, понятие типа данных и строгой типизации; стандартные типы языка Java (примитивные и String), преобразования типов; особенности хранения различных типов в памяти компьютера и их обработки; System.in и System.out, ввод данных с помощью класс java.util.Scanner, форматирование вывода с помощью printf и String.format	
2.2	Функции	Понятие функций (статических методов класса применительно к Java): описание и вызов, передача параметров; проектирование программы с использованием функций; важность разделения программы на подзадачи и превильного выделения подзадач, структуризация и принципы повторного использования кода; примеры программ с использованием функций	
2.3	Операторы управления ходом выполнения программ	Условный оператор, составной оператор, тернарный оператор, операторы циклов в Java, операторы break и continue; вложенные блоки кода и видимость пеменных; соглашения по оформлению Java-кода; примеры решения задач	
2.4	Простейшие алгоритмы	Примеры решения задач: перевод десятичного числа в двоичное представление, собственная реализация sqrt методом половинного деления, вычисление числа Пи, выделение n-ой части строки, разделенной запятыми, печать n символов последовательности и др.; разные варианты решения задач и выбор оптимального варианта	
2.5	Составные типы данных	Массивы и множественные массивы (массивы массивов); типичные задачи обработки массивов, класс java.util.Arrays; разработка библиотеки функций ввода/вывода массивов и других функции в виде класс ArrayUtils. Перечисления (enum). Структыры данных в виде простейших классов, важность применения структур для упрощения и струкутризация кода, массивы структур, примеры задач. Массивы и структуры в качестве параметров функций и возвращаемых значений. Понятие типов-значений (value types) и ссылочных типов (reference types) данных в Java, понятие объектов, ссылочная модель и сборка мусора.	
2.6	Строки	Строки в Java, особенности класса String - особенности реализации (неизменяемость), поддерживаемые методы, сравнение строк с помощью equals. Особенности конкатенации строк и класс StringBuilder. Поверхностное знакомство с регулярными выражения и возможностями их применения (RegExp и некоторые методы String).	

2.7	Типичные задачи обработки массивов и строк	Примеры решения задач: поиск минимума/максимума в массиве, поиск индекса элемента, сортировка массива методом "пузырька", передача различных критериев сортировки в метод Arrays.sort, бинарный поиск в упорябоченном массиве, операции со столбцами и строка в двумерном массиве и др.; демострации важности структуризации программы с помощью фунций на примере задач обработки массивов и строк	
2.8	Основы объектно-ориентированного подхода	Принципы объектно ориентированного подхода; понятие класса и его экземпляров; различия между static-функциями и методами классов; классы как расширение концепции типа данных в виде объединения данных и методов их обработки, понятие состояния объекта, примеры; принципы инкапсуляции; понятие наследования и полиморфизма; класс Object и его методы	
2.9	Создание приложений с оконным интерфейс	Принципы создания приложений с оконным интерфейсом; библиотека Swing и базовые Swing-компоненты; визуальное проектирование форм (JFrame) в среде разработки, обработка событий компонентов; понятие модели данных для сложных копонентов, JTable и разработка класса JTableUtils для упрощения работы с JTable. Типичная архитектура приложений с оконным интерфейсом и важность разделения логики и отображения, приложения с несколькими формами, примеры приложений. Второй вариант построения оконного интерфейса - с помощью JavaFX, возможности и особенности, примеры приложений.	
2.10	Коллекции	Понятие коллекций, какие виды колдекций бывают (списки, словари, множества, стеки и очереди), иерархия коллекций в Java; списки (List<T>) по сравнению с массивами, методы списков; словари (Map<K, V>) и множества (Set<T>) и их методы; понятие обобщенных типов данных и кода (generics); разница между интерфейсом и реализацией (List<T> и ArrayList<T>/LinkedList<T>, Map<K, V> и TreeMap<K, V>/HashMap<K, V>); примеры эффективного решения задач с помощью коллекций	
2.11	Создание прикладных приложений (создание игры)	Пример создания полноценного оконного приложения на Java - игры "Сапер" в качестве демонстрации применения структур данных (классов, массивов) и алгоритмов применительно к конкретной практической задаче; архитектура приложения с применение ООП-подхода - выделение логики в виде отдельно класса, реализация отображение в JTable; знакомство с классом java.awt.Graphics	
2.12	Рекурсия и рекурсивные алгоритмы	Понятие рекурсии в программирование; прямая и косвенная рекурсия; вычисление факториала, когда стоит и когда не стоит использовать рекурсию; рекурсивное вычисление чисел Фибоначчи - экспоненциальный рост количества повторных вызовов функций в некоторых рекурсивных алгоритмах и возможное решения с помощью кеширования результатов вычислений; примеры решения задач: рисование треугольника Серпинского, сопоставление строки шаблону, обход двумерного поля в глубину и ширину; варианты реализации истинно рекурсивных алгоритмов без рекурсивных вызовов с применение стеков (Stack<T>) и очередей (Queue<T>)	

3. Лабораторные работы			
3.1	Переменные и типы данных, ввод-вывод данных	Понятие переменной, понятие типа данных и строгой типизации; стандартные типы языка Java (примитивные и String), преобразования типов; особенности хранения различных типов в памяти компьютера и их обработки; System.in и System.out, ввод данных с помощью класс java.util.Scanner, форматирование вывода с помощью printf и String.format	
3.2	Функции	Понятие функций (статических методов класса применительно к Java): описание и вызов, передача параметров; проектирование программы с использованием функций; важность разделения программы на подзадачи и превильного выделения подзадач, структуризация и принципы повторного использования кода; примеры программ с использованием функций	
3.3	Операторы управления ходом выполнения программ	Условный оператор, составной оператор, тернарный оператор, операторы циклов в Java, операторы break и continue; вложенные блоки кода и видимость пеменных; соглашения по оформлению Java-кода; примеры решения задач	
3.4	Простейшие алгоритмы	Примеры решения задач: перевод десятичного числа в двоичное представление, собственная реализация sqrt методом половинного деления, вычисление числа Пи, выделение n-ой части строки, разделенной запятыми, печать n символов последовательности и др.; разные варианты решения задач и выбор оптимального варианта	
3.5	Составные типы данных	Массивы и множественные массивы (массивы массивов); типичные задачи обработки массивов, класс java.util.Arrays; разработка библиотеки функций ввода/вывода массивов и других функции в виде класс ArrayUtils. Перечисления (enum). Структыры данных в виде простейших классов, важность применения структур для упрощения и струкутризация кода, массивы структур, примеры задач. Массивы и структуры в качестве параметров функций и возвращаемых значений. Понятие типов-значений (value types) и ссылочных типов (reference types) данных в Java, понятие объектов, ссылочная модель и сборка мусора.	
3.6	Строки	Строки в Java, особенности класса String - особенности реализации (неизменяемость), поддерживаемые методы, сравнение строк с помощью equals. Особенности конкатенации строк и класс StringBuilder. Поверхностное знакомство с регулярными выражения и возможностями их применения (RegExp и некоторые методы String).	
3.7	Типичные задачи обработки массивов и строк	Примеры решения задач: поиск минимума/максимума в массиве, поиск индекса элемента, сортировка массива методом "пузырька", передача различных критериев сортировки в метод Arrays.sort, бинарный поиск в упорябоченном массиве, операции со столбцами и строка в двумерном массиве и др.; демострации важности структуризации программы с помощью фунций на примере задач обработки массивов и строк	

3.8	Основы объектно-ориентированного подхода	Принципы объектно ориентированного подхода; понятие класса и его экземпляров; различия между static-функциями и методами классов; классы как расширение концепции типа данных в виде объединения данных и методов их обработки, понятие состояния объекта, примеры; принципы инкапсуляции; понятие наследования и полиморфизма; класс Object и его методы	
3.9	Создание приложений с оконным интерфейс	Принципы создания приложений с оконным интерфейсом; библиотека Swing и базовые Swing-компоненты; визуальное проектирование форм (JFrame) в среде разработки, обработка событий компонентов; понятие модели данных для сложных копонентов, JTable и разработка класса JTableUtils для упрощения работы с JTable. Типичная архитектура приложений с оконным интерфейсом и важность разделения логики и отображения, приложения с несколькими формами, примеры приложений. Второй вариант построения оконного интерфейса - с помощью JavaFX, возможности и особенности, примеры приложений.	
3.10	Коллекции	Понятие коллекций, какие виды колдекций бывают (списки, словари, множества, стеки и очереди), иерархия коллекций в Java; списки (List<T>) по сравнению с массивами, методы списков; словари (Map<K, V>) и множества (Set<T>) и их методы; понятие обобщенных типов данных и кода (generics); разница между интерфейсом и реализацией (List<T> и ArrayList<T>/LinkedList<T>, Map<K, V> и TreeMap<K, V>/HashMap<K, V>); примеры эффективного решения задач с помощью коллекций	
3.11	Создание прикладных приложений (создание игры)	Пример создания полноценного оконного приложения на Java - игры "Сапер" в качестве демонстрации применения структур данных (классов, массивов) и алгоритмов применительно к конкретной практической задаче; архитектура приложения с применение ООП-подхода - выделение логики в виде отдельно класса, реализация отображение в JTable; знакомство с классом java.awt.Graphics	
3.12	Рекурсия и рекурсивные алгоритмы	Понятие рекурсии в программирование; прямая и косвенная рекурсия; вычисление факториала, когда стоит и когда не стоит использовать рекурсию; рекурсивное вычисление чисел Фибоначчи - экспоненциальный рост количества повторных вызовов функций в некоторых рекурсивных алгоритмах и возможное решения с помощью кеширования результатов вычислений; примеры решения задач: рисование треугольника Серпинского, сопоставление строки шаблону, обход двумерного поля в глубину и ширину; варианты реализации истинно рекурсивных алгоритмов без рекурсивных вызовов с применение стеков (Stack<T>) и очередей (Queue<T>)	

13.2. Темы (разделы) дисциплины и виды занятий

№ п/п	Наименование темы (раздела) дисциплины	Виды занятий (часов)				
		Лекции	Практи-ческие	Лабора-торные	Самостоя-тельная работа	Всего
1	Введение в предмет	2	–	–	2	4
2	Переменные и типы данных, ввод-вывод данных	2	1	1	4	8
3	Функции	3	1	1	2	7

4	Операторы управления ходом выполнения программ	3	2	2	4	11
5	Простейшие алгоритмы	2	1	1	2	6
6	Составные типы данных	3	1	1	4	9
7	Строки	2	1	1	2	6
8	Типичные задачи обработки массивов и строк	2	2	2	4	10
9	Основы объектно-ориентированного подхода	3	1	1	4	9
10	Создание приложений с оконным интерфейс	4	2	2	4	12
11	Коллекции	2	1	1	2	6
12	Создание прикладных приложений (создание игры)	3	2	2	6	13
13	Рекурсия и рекурсивные алгоритмы	3	1	1	2	7
	Итого:	34	16	16	40	108

14. Методические указания для обучающихся по освоению дисциплины

Рекомендуется работа с конспектами лекций, презентационным материалом, выполнение всех лабораторных и контрольных работ, заданий текущей аттестации. Учебные и методические материалы по дисциплине размещены на сетевом диске, доступным на любом компьютере в локальной сети ФКН.

15. Перечень основной и дополнительной литературы, ресурсов интернет, необходимых для освоения дисциплины

а) основная литература:

№ п/п	Источник
1	Эккель, Брюс. Философия Java = Thinking in Java / Брюс Эккель ; [пер. с англ. Е. Матвеева] .— 4-е полное изд. — Москва : Вильямс, 2017 .— 1165 с. : ил. — (Классика computer science) .— ISBN 978-5-496-01127-3.
2	Хорстманн, Кей. Java = Core Java / Кей Хорстманн ; [пер. с англ. и ред. И.В. Берштейна] .— Москва ; Санкт-Петербург ; Киев : Вильямс, 2017- .— (Библиотека профессионала) .— ISBN 978-5-8459-2083-6.
3	Блох, Джошуа. Java. Эффективное программирование = Effective Java programming language guide / Джошуа Блох ; пер. с англ. В. Стрельцов ; науч. ред. Р. Усманов ; предисл. Г. Стила .— Москва : Лори, 2017 .— 294 с. : табл. — (Серия Java "...из первых рук") .— Библиогр.: с. 288-294 .— ISBN 978-5-85582-347-9.

б) дополнительная литература:

№ п/п	Источник
4	Хорстманн, Кей. Java = Core Java / К. Хорстманн, Г. Корнелл ; [пер. с англ. и ред.И.В. Берштейна] .— Москва ; Санкт-Петербург ; Киев : Вильямс, 2015 .— (Библиотека профессионала) .— ISBN 978-5-8459-2032-4.
5	Шилдт, Герберт. Java : руководство для начинающих / Герберт Шилдт ; [пер. с англ. и ред.В.В. Вейтмана] .— 4-е изд. — М. [и др.] : Вильямс, 2009 .— 715 с. : ил. — Предм. указ.: с.709-715 .— ISBN 978-5-8459-1440-8.
6	Портянкин, Иван Александрович. Swing. Эффектные пользовательские интерфейсы. Java Foundftlon Classos / Иван Портянкин .— СПб. [и др.] : Питер, 2005 .— 523 с. — (Библиотека программиста) .— Алф. указ.: с.515-523 .— ISBN 5-469-00005-2.
7	Дейтел, Х.М. Как программировать на Java / Х.М. Дейтел, П.Д. Дейтел .— М. : Бином, 2003-.Кн. 1: Основы программирования / Пер. с англ. под ред. А.В. Козлова .— 4-е изд. — 2003 .— 847 с. : ил. — Парал. тит. л. англ. — ISBN 5-9518-0015-3.
8	Технологии программирования на Java 2 / Х.М. Дейтел, П.Д. Дейтел, С.И. Сантри .— М. : Бином, 2003-.Кн. 1: Графика, JavaBeans, интерфейс пользователя / Пер. с англ. под. ред. А.И. Тихонова .— 2003 .— 560 с. : ил. — Парал. тит. л. англ. — ISBN 5-9518-0017-X .— ISBN 0-13-089560-1.

в) информационные электронно-образовательные ресурсы (официальные ресурсы интернет):

№ п/п	Источник
9	Самоучитель по Java с нуля [Электронный ресурс] : — Режим доступа: https://vertex-academy.com/tutorials/ru/samouchitel-po-java-s-nulya/
10	Учебник: программирование на Java [Электронный ресурс] : — Режим доступа: https://java9.ru/
11	Иллюстрированный самоучитель по Java [Электронный ресурс] : — Режим доступа: http://www.realcoding.net/teach/java/

16. Перечень учебно-методического обеспечения для самостоятельной работы

№ п/п	Источник
1	Шилдт, Герберт. Искусство программирования на JAVA : пер. с англ. / Герберт Шилдт, Джеймс Холмс .— СПб. [и др.] : БХВ-Петербург, 2005 .— 331 с. : ил. — Парал. тит. л. англ. — Предм. указ. : с.330-331, 4000 экз.
2	Лафоре, Роберт. Структуры данных и алгоритмы в Java = Data structures @ algorithms in Java / Роберт Лафоре ; [пер. с англ. Е. Матвеева] .— 2-е изд. — Санкт-Петербург [и др.] : Питер, 2014 .— 701 с. : ил., табл. — (Классика computer science) .— Библиогр.: с.683-685 .— Алф. указ.: с.695-701 .— ISBN 985-5-496-00740-5.

17. Информационные технологии, используемые для реализации учебной дисциплины, включая программное обеспечение и информационно-справочные системы (при необходимости)

№ п/п	Наименование
1	OpenJDK - беплатен
2	Среда разработки NetBeans или Intellij IDEA (академическая лицензия или версия Community) - бесплатны

18. Материально-техническое обеспечение дисциплины:

№ п/п	Наименование
1	Мультимедийная лекционная аудитория (корп. 1а, ауд. № 479 или другая подходящая): рабочее место преподавателя: ПК-Intel-i3, проектор, видеокоммутатор, микрофон, аудиосистема, специализированная мебель: доски меловые 2 шт., столы и стулья/лавки в количестве, достаточном для размещения потока студентов; выход в Интернет, доступ к фондам учебно-методической документации и электронным изданиям.
2	Компьютерный класс (корп. 1а, ауд. № 382-385 или другие подходящие): ПК-Intel-i3 16 шт., специализированная мебель: доска маркерная 1 шт., столы и стулья в количестве, достаточном для размещения академической группы (подгруппы) студентов; выход в Интернет, доступ к фондам учебно-методической документации и электронным изданиям.

19. Оценочные средства для проведения текущей и промежуточной аттестаций

Порядок оценки освоения обучающимися учебного материала определяется содержанием следующих разделов дисциплины:

№ п/п	Наименование раздела дисциплины (модуля)	Компетен-ция(и)	Индикатор(ы) достижения компетенции	Оценочные средства
1	Введение в предмет	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
2	Переменные и типы данных, ввод-вывод данных	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
3	Функции	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)

4	Операторы управления ходом выполнения программ	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
5	Простейшие алгоритмы	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
6	Составные типы данных	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
7	Строки	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
8	Типичные задачи обработки массивов и строк	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
9	Основы объектно-ориентированного подхода	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
10	Создание приложений с оконным интерфейс	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
11	Коллекции	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
12	Создание прикладных приложений (создание игры)	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
13	Рекурсия и рекурсивные алгоритмы	ОПК-7	ОПК-7.1, ОПК-7.2, ОПК-7.3, ОПК-7.4, ОПК-7.5, ОПК-7.6	Обязательные практические задания из пункта 20.1 (контроль и оценка выполнения)
Промежуточная аттестация форма контроля – экзамен				Перечень вопросов к экзамену из пункта 20.2

20. Типовые оценочные средства и методические материалы, определяющие процедуры оценивания

20.1 Текущий контроль успеваемости

Контроль успеваемости по дисциплине осуществляется с помощью контроля выполнения обязательных практических заданий. Перечень заданий:

№ п/п	Задание
1	Задача 1 - Запись выражений и оператор присвания (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
2	Задача 2 - Условный оператор (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
3	Задача 3 - Примененение функций (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
4	Задача 4 - Циклы (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
5	Задача 5 - Циклы (рисование фигуры псевдографикой, >= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
6	Задача 6 - Циклы (вычисление суммы ряда, >= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
7	Задача 7 - Одномерные массивы (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
8	Задача 8 - Двумерные массивы (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
9	Задача 9 - Коллекции (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)

10	Задача 10 - Структуры данных (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
11	Задача 11 - Строки (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
12	Задача 12 - Рекурсия (>= 20 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)
13	Задача 13 - Логическая игра (>= 30 индивидуальных вариантов, размещены на общедоступном диске в сети ФКН)

20.2 Промежуточная аттестация

Для оценивания результатов обучения на зачете используются следующие содержательные показатели (формулируется с учетом конкретных требований дисциплины):

1) знание теоретических основ учебного материала, основных определений, понятий и используемой терминологии;

2) умение проводить обоснование и представление основных теоретических и практических результатов (теорем, алгоритмов, методик) с использованием математических выкладок, блок-схем, структурных схем и стандартных описаний к ним;

3) умение связывать теорию с практикой, иллюстрировать ответ примерами, в том числе, собственными, умение выявлять и анализировать основные закономерности, полученные, в том числе, в ходе выполнения лабораторно-практических заданий;

4) умение обосновывать свои суждения и профессиональную позицию по излагаемому вопросу;

5) владение навыками программирования и экспериментирования в рамках выполняемых лабораторных заданий;

Различные комбинации перечисленных показателей определяют критерии оценивания результатов обучения (сформированности компетенций) на зачете:

– высокий (углубленный) уровень сформированности компетенций;

– повышенный (продвинутый) уровень сформированности компетенций;

– пороговый (базовый) уровень сформированности компетенций.

Для оценивания результатов обучения на зачете с оценкой используется 4-балльная шкала: «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Соотношение показателей, критериев и шкалы оценивания результатов обучения на экзамене представлено в следующей таблице.

Критерии оценивания компетенций	Уровень сформированности компетенций	Шкала оценок
Студент владеет основными понятиями учебной дисциплины, может пояснить большинство принципов на примерах; вовремя сдал все практические задания, которые выполнены на высоком уровне, без явных ошибок.	Повышенный уровень	Отлично
Студент владеет основными понятиями учебной дисциплины, однако в ответах на некоторые вопросы допускает неточности; сдал все практические задания, однако к некоторым решениям студента у преподавателя есть замечания.	Базовый уровень	Хорошо
Студент знает основные определения из учебной дисциплины, однако пояснить многие понятия на примерах затрудняется; сдал большую часть практических заданий, однако продемонстрированные решения содержат существенные ошибки.	Пороговый уровень	Удовлетворительно
Студент путается в основных понятиях учебной дисциплины, не может привести примеры; не сдал большую часть практических заданий.	–	Неудовлетвори-тельно

Перечень вопросов к экзамену (зачету):

№ п/п	Вопрос
1	Алгоритм и его свойства
2	Обзор языка Java
3	Типы данных языка Java: типы-значения и ссылочные типы, обзор числовых типов

4	Переменнные, область видимости переменных, строгая типизация
5	Строки и работа со строками (обзор String, StringBuilder, RegExp)
6	Операторы языка Java
7	Соглашения по оформлению Java-кода
8	Функции, структуризация программ с помощью функций
9	Ввод-вывод данных в Java
10	Массивы: одномерные, двумерные, типичные задачи с использованием массивов
11	Сортировка: реализая пузыльковой сортировки, Arrays.sort, различные критерии сортировки
12	Поиск в массиве: последовательный поиск, бинарный в отсортированном массиве
13	Составные типы данных - классы
14	Основы объекно-ориентиртованного подхода
15	Коллекции в языке Java
16	Практические примеры работы со словарями (Map)
17	Обобщенное программирование (generics), классы-обертки над примитвными типами данных
18	Построение оконного интерфейса с помощью библиотеки Swing, архитектура приложения
19	Работа с компонентом JTable
20	"Рисование" в Java, обзор методов класса Graphics
21	Обзор библиотеки и принципов создания приложений JavaFX
22	Рекурсия и рекурсивные алгоритмы
23	Рекурсивные алгоритмы: сопоставление строки шаблону
24	Рекурсивные алгоритмы: рисование фракталов
25	Рекурсивные алгоритмы: обход поля в глубину
26	Реализация рекурсивных алгоритмов без рекурсии с помощью стека и очереди

20.3. Приведённые ниже задания рекомендуется использовать при проведении диагностических работ для оценки остаточных знаний по дисциплине

Вопросы с множественным ответом (5)

1. Строгая типизация предполагает (выберите верные утверждения)?
   - Все используемые в функции переменные должны объявляться строго до остального кода функции.
   - При компиляции программы весь код (все операции) проверяется на совместимость или возможность преобразования типов, несовместимость считается ошибкой.
   - В программе нельзя определить несколько функций с одинаковым именем.
   - Язык программирования обязательно должен быть объектно-ориентированным.

2. Какие утверждения верны для массивов в языке Java?
   - Размер массива может быть изменен после его создания.
   - Индексация элементов в массиве начинается с 1.
   - Все элементы в конкретном массиве должны быть одного типа (или наследоваться от одного типа).
   - В одной программе могут использоваться массивы только для одного типа данных.

3. Почему для конкатенации множества строк в языке Java следует использовать StringBuilder (выберите верные утверждения)?
   - Конкатенация строк оператором «+» не предусмотрена.
   - При конкатенации строк с помощью оператором «+» результат всегда печатается в консоль (стандартный поток вывода – stdout).
   - Конкатенация строк оператором «+» приводит к созданию множества экземпляров строк и многократному копированию данных.
   - Строки не являются ссылочным типом данных.

4. Что возвращает функция, приведенная ниже:

```java
public int getXYZ(int[] arr) {
    int a = -1;
    for (int i = 0; i < arr.length; i++) {
        if (arr[i] > 0 && (a < 0 || arr[i] < a)) {
            a = arr[i];
        }
    }
    return a;
}
```

- последнее положительное значение в массиве;
- максимальное значение в массиве;
- минимальное значение после первого положительного значения в массиве;
- минимальное положительное значение в массиве.

5. Какие числа встретятся среди всех напечатанных при выполнении следующего фрагмента кода (требуется указать все правильные варианты):

```java
int a = 2, b = 0;
for (int i = 0; i < 20; i++) {
    System.out.println(a);
    a += 7;
    if (a >= 20) {
        a -= 20;
        b++;
        if (b >= 3)
            break;
    }
}
```

- 15
- 16
- 17
- 18

6. Строгая типизация предполагает (выберите верные утверждения)?
   - Все используемые в функции переменные должны объявляться строго до остального кода функции.
   - При компиляции программы весь код (все операции) проверяется на совместимость или возможность преобразования типов, несовместимость считается ошибкой.
   - В программе нельзя определить несколько функций с одинаковым именем.
   - Язык программирования обязательно должен быть объектно-ориентированным.

7. Определите, что вычисляет следующая функция:

```java
public static int solve(int[] arr) {
    int minIndex = 0;
    int maxIndex = 0;
    for (int i = 1; i < arr.length; i++) {
        if (arr[i] < arr[minIndex]) {
            minIndex = i;
        }
        if (arr[i] >= arr[maxIndex]) {
            maxIndex = i;
        }
    }
    return Math.max(Math.abs(minIndex - maxIndex) - 1, 0);
}
```

- количество элементов в массиве между первым минимальным и последним максимальным значением;
- разность между минимальных и максимальным элементом массива;
- количество элементов в массиве между первым максимальным и последним минимальным значением;
- разность между общим количеством минимальных и максимальных элементов в массиве.

8. Выберите верные утверждения для циклов:

- Цикл for выполняется в 2 раза быстрее, чем цикл while.
- Циклы не могут быть использованы в рекурсивных функциях.
- В C-подобных языках (например, Java) любой цикл while формально может быть переписан в виде цикла for.
- Количество вложенных циклов в функции не может быть больше, чем количество параметров в этой функции.

Вопросы с коротким ответом (1)

1. Какая строка будет напечатана в результате выполнения следующего кода?

```java
char ch = 'a';
int a = 256;
int b = 0;
while (a > b) {
    System.out.print(ch);
    ch++;
    a = a / 2;
    b++;
}
System.out.println();
```

Ответ: abcdef

2. Какое максимальное значение могло храниться в переменной x, если в результате выполнения кода, приведенного ниже, было напечатано число 12?

```
int[] arr = {1, 3, x, 7, 10};
int s = 0;
for (int i = 0; i < arr.length; i++) {
    s += arr[i];
}
s = s / 4;
System.out.println(s);
```

Ответ: 30

Вопросы развернутые (1)

1. Реализовать на любом языке программирования наиболее эффективный алгоритм, который в массиве целых чисел перенесет в начало массива все нулевые элементы, при этом взаимное расположение ненулевых элементов не должно измениться.

Пример работы алгоритма, который требуется реализовать:
[1, 0, 0, 3, 2, 0, 6, 0, 0, 4, 5] → [0, 0, 0, 0, 0, 1, 3, 2, 6, 4, 5]

Ответ:

```
public void task(int[] arr) {
    int j = arr.length - 1;
    for (int i = arr.length - 1; i >= 0; i--) {
        if (arr[i] != 0) {
            if (i < j) {
                arr[j] = arr[i];
            }
            j--;
        }
    }
    for (int i = 0; i <= j; i++) {
        arr[i] = 0;
    }
}
```

Критерии оценивания ответа

Критерии оценивания	Шкала оценок
Обучающийся приводит полную и безошибочную реализацию алгоритма, который работает за O(n).	Отлично (90-100 баллов)
Обучающий приводит корректную реализацию алгоритма, который работает за время, большее, чем O(n).	Хорошо (70-80 баллов)

В реализации допущены некоторые ошибки, но в целом идея реализации алгоритма правильная.	Удовлетворительно (50-70 баллов)
Приведенный алгоритм содержит грубые ошибки реализации.	Неудовлетворительно (менее 50 баллов)

2. Реализовать на любом языке программирования алгоритм, который в массиве целых чисел найдет значение максимального локального минимума. Под локальный минимумом понимается элемент, соседи которого больше данного элемента (у первого и последнего элемента – по одному соседу, у всех остальных – по два, слева и справа).

Пример работы алгоритма, который требуется реализовать (цветом выделен локальные минимумы):

[**1**, 3, 4, 3, 3, 5, 8, **4**, 5, **3**, 5] → 4

Ответ:

```
public static int solve(int[] arr) {
    int max = Integer.MIN_VALUE;
    for (int i = 0; i < arr.length; i++) {
        if ((i == 0 || arr[i] < arr[i - 1]) &&
            (i == arr.length - 1 || arr[i] < arr[i + 1])) {
            max = Math.max(arr[i], max);
        }
    }
    return max;
}
```

Критерии оценивания ответа

Критерии оценивания	Шкала оценок
Обучающийся приводит полную и безошибочную реализацию алгоритма, который работает за O(n), возможно, с выделением функции проверки элемента на локальный минимум.	Отлично (90-100 баллов)
Обучающий приводит корректную реализацию алгоритма, однако более запутанную, чем возможно.	Хорошо (70-80 баллов)

В реализации допущены некоторые ошибки, но в целом идея реализации алгоритма правильная.	Удовлетворительно (50-70 баллов)
Приведенный алгоритм содержит грубые ошибки реализации.	Неудовлетворительно (менее 50 баллов)