МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования

«ИРКУТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных технологий
Кафедра информационных технологий

«УТВЕРЖДАЮ»
Директор ИМИТ ИГУ
_____ М. В. Фалалеев
«29» апреля 2026 г.

Рабочая программа дисциплины (модуля)

Б1.О.13 Программирование

Направление подготовки 02.03.02 Фундаментальная информатика и информационные технологии

Направленность (профиль) подготовки Фундаментальная информатика и программная инженерия

Квалификация выпускника бакалавр

Форма обучения очная

Иркутск 2026 г.

1. ЦЕЛИ И ЗАДАЧИ ДИСЦИПЛИНЫ

Цель

Формирование у студентов базовых знаний в области информатики и программирования.

Задачи:

Формирование знаний, умений и навыков студента по разделам «Массивы и строки», «Рекурсия», «Объектно-ориентированное программирование», «Коллекции в Java», формирование базовых практических умений и навыков программирования на языке Java

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОПОП ВО

2.1. Учебная дисциплина (модуль) относится к обязательной части программы и изучается на первом курсе.

2.2. Для изучения данной учебной дисциплины (модуля) необходимы знания, умения и навыки, сформированные: Информатика.

2.3. Перечень последующих учебных дисциплин, для которых необходимы знания, умения и навыки, формируемые данной учебной дисциплиной: Введение в информационный поиск, Проектирование информационных систем.

3. ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ОСВОЕНИЯ ДИСЦИПЛИНЫ

Процесс освоения дисциплины направлен на формирование компетенций (элементов следующих компетенций) в соответствии с ФГОС ВО по соответствующему направлению подготовки.

Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с индикаторами достижения компетенций

Компетенция	Индикаторы компетенций	Результаты обучения
ОПК-2 Способен применять компьютерные/суперкомпьютерные методы, современное программное обеспечение, в том числе отечественного происхождения, для решения задач профессиональной деятельности	ИДК $_{ОПК2.1}$ Понимает базовые принципы и устройство современных информационных технологий и программных средств	Знает основные способы и принципы представления структур данных и приемы алгоритмизации Умеет формализовать поставленную задачу, составлять и оформлять программы на языке программирования Java Владеет навыком распознавания обобщенных приемов и методов решения типовых классов задач
	ИДК $_{ОПК2.2}$ Способен применять современное программное обеспечение (в том числе отечественного производства) для решения задач профессиональной деятельности	Знает Java Core. Умеет работать в современных средах разработки для написания программного кода на Java, тестировать и отлаживать программы в современных интегрированных средах разработки.

		Владеет навыками разработки, отладки и тестирования программ в интегрированной среде разработки.
	ИДК $_{ОПК2.3}$ Способен применять суперкомпьютерные методы для решения задач профессиональной деятельности	Знает Java Core. Умеет работать в современных средах разработки для написания программного кода на Java Владеет навыками разработки, отладки и тестирования программ в интегрированной среде разработки.
ОПК-3 Способен к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям	ИДК $_{ОПК3.1}$ Знает основные языки программирования и типы баз данных, операционные системы и оболочки, современные программные среды разработки информационных систем и технологий	Знает Java Core. Умеет работать в современных средах разработки для написания программного кода на Java, тестировать и отлаживать программы в современных интегрированных средах разработки. Владеет навыками разработки, отладки и тестирования программ в интегрированной среде разработки.
	ИДК $_{ОПК3.2}$ Применяет языки программирования и современные программные среды разработки информационных систем и технологий для автоматизации бизнес-процессов, решения прикладных задач различных классов, создания информационных ресурсов глобальных сетей, ведения баз данных и информационных хранилищ	Знает основные способы и принципы представления структур данных и приемы алгоритмизации, принципы структурного и модульного программирования, принципы объектно-ориентированного программирования, принципы разработки программ; Умеет формализовать поставленную задачу, составлять и оформлять программы на языке программирования Java, применять полученные знания к различным предметным областям, разрабатывать и описывать классы объектов разного целевого характера, создавать иерархии классов Владеет приемами работы с учебной, научной, справочной литературой, навыком распознавания обобщенных приемов и методов решения

		типовых классов задач
	ИДК ОПК3.3 Способен выполнять задачи программирования, отладки и тестирования прототипов программных средств и информационных систем	Знает приемы работы с учебной, научной, справочной литературой, навыком распознавания обобщенных приемов и методов решения типовых классов задач Умеет составлять и оформлять программы на языке программирования Java, тестировать и отлаживать программы в современных интегрированных средах разработки Владеет навыками разработки, отладки и тестирования программ в интегрированной среде разработки

4. СОДЕРЖАНИЕ И СТРУКТУРА ДИСЦИПЛИНЫ

Объем дисциплины составляет 4 зачетных единицы, 144 часа, в том числе 35 часов на контроль, практическая подготовка 144.
Форма промежуточной аттестации: 2 семестр – экзамен.

4.1. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ, СТРУКТУРИРОВАННОЕ ПО ТЕМАМ, С УКАЗАНИЕМ ВИДОВ УЧЕБНЫХ ЗАНЯТИЙ И ОТВЕДЕННОГО НА НИХ КОЛИЧЕСТВА АКАДЕМИЧЕСКИХ ЧАСОВ

№ п/п	Раздел дисциплины/темы	Семестр	Виды учебной работы, включая самостоятельную работу обучающихся и трудоемкость (в часах)					Формы текущего контроля успеваемости
			Контактная работа преподавателя с обучающимися			Самостоятельная работа + контроль		
			Лекции	Семинарские (практические занятия)	Контроль обучения			
1	Тема 1. Класс StdDraw	2	2	2	1,25	2	4,375	лаб.
2	Тема 2. Массивы	2	4	8	1,25	2	4,375	лаб.
3	Тема 3. Символьный тип данных. Строки	2	2	3	1,25	2	4,375	лаб.
4	Тема 4. Рекурсия	2	4	3	1,25	4	4,375	лаб.
5	Тема 5. Объектно-ориентированное программирование	2	14	14	1,25	8	4,375	лаб.
6	Тема 6. Обработка исключений	2	1	1	1,25	2	4,375	лаб.
7	Тема 7. Стандарт документирования Javadoc	2	1	1	1,25	1	4,375	лаб.

8	Тема 8. Коллекции в Java	2	8	4	1,25	6	4,375	лаб.
Итого часов			36	36	10	62		

4.2. ПЛАН ВНЕАУДИТОРНОЙ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

Семестр	Название раздела, темы	Самостоятельная работа обучающихся			Оценочное средство	Учебно-методическое обеспечение самостоятельной работы
		Вид самостоятельной работы	Сроки выполнения	Затраты времени (час.)		
2	Тема 1. Класс StdDraw	Выполнение практической работы	24.02	6,375	Домашняя работа	Литература из п. 5 и материалы курса на платформе ИОС DOMIC
2	Тема 2. Массивы	Выполнение практической работы	24.03	6,375	Домашняя работа	Литература из п. 5 и материалы курса на платформе ИОС DOMIC
2	Тема 3. Символьный тип данных. Строки	Выполнение практической работы	07.04	6,375	Домашняя работа	Литература из п. 5 и материалы курса на платформе ИОС DOMIC
2	Тема 4. Рекурсия	Выполнение практической работы	14.04	8,375	Домашняя работа	Литература из п. 5 и материалы курса на платформе ИОС DOMIC
2	Тема 5. Объектно-ориентированное программирование	Выполнение практической работы	02.06	12,375	Домашняя работа	Литература из п. 5 и материалы курса на платформе ИОС DOMIC
2	Тема 6. Обработка исключений	Выполнение практической работы	09.06	6,375	Домашняя работа	Литература из п. 5 и материалы курса на платформе ИОС DOMIC
2	Тема 7. Стандарт документирования Javadoc	Выполнение практической работы	09.06	5,375	Домашняя работа	Литература из п. 5 и материалы курса на платформе ИОС DOMIC
2	Тема 8. Коллекции в Java	Выполнение практической работы	09.06	10,375	Домашняя работа	Литература из п. 5 и материалы курса на платформе ИОС DOMIC
Общая трудоемкость самостоятельной работы по дисциплине (час)				62		
Из них объем самостоятельной работы с использованием электронного обучения и дистанционных образовательных технологий (час)				62		

4.3. СОДЕРЖАНИЕ УЧЕБНОГО МАТЕРИАЛА

1. Тема 1. Класс StdDraw

Работа с системой координат, управление цветом. Рисование графических примитивов: точки, отрезки, прямоугольники, окружности, многоугольники. Построение графиков функций. Создание анимации с помощью StdDraw.

Тема 2. Массивы

Сортировка элементов массива, алгоритмы сортировки, метод пузырька, метод пузырька с флажком, метод выбора, формирование массива по условию, поиск в массиве, линейный поиск, двоичный поиск, массив как параметр метода, многомерные массивы (матрицы), объявление многомерного массива, ввод и вывод, обработка всех элементов массива, операции с матрицами, неоднородные (нерегулярные) массивы, инициализация многомерных массивов. Методы класса Arrays.

Тема 3. Символьный тип данных. Строки

Кодирование символов, однобайтовые кодировки, кодировка Unicode, символьный тип данных, массив символов, символьные строки, ввод с клавиатуры, указание кодировки, методы класса String. Работа с классом StringBuffer.

Тема 4. Рекурсия

Рекурсия, рекурсивные объекты, рекурсия и математическая индукция, основное правило рекурсии, рекурсия и итерация, вычисление факториала числа, вычисление чисел Фибоначчи, динамическое программирование.

Тема 5. Объектно-ориентированное программирование

Объектно-ориентированное программирование (ООП), основные принципы ООП, объекты, состояние и поведение объекта, класс, общая форма класса, члены класса, аргументы экземпляра, методы, пример класса Box, объявление объектов, присваивание переменным объектных ссылок, общая форма объявления метода, вызов метода объекта, метод с параметрами, конструктор класса, особенности конструкторов, конструктор с параметрами, ключевое слово this, сокрытие переменной экземпляра. Наследование, суперкласс, подкласс, общая форма наследование, доступ к членам класса и наследование, ссылочная переменная суперкласса, ключевое слово super, вызов конструктора суперкласса, обращение к члену суперкласса, класс Object, методы класса Object, порядок вызова конструкторов, конструктор по умолчанию, затенение полей родительского класса, переопределение методов, динамическая диспетчеризация методов, абстрактные методы, абстрактные классы, особенности абстрактных классов, ключевое слово final и наследование, предотвращение перегрузки метода, предотвращение наследования класса, пример наследования: абстрактный класс Triad, подклассы Time, Date, класс Memory. Ограничение доступа, публичные, закрытые, защищенные члены класса, доступность членов класса, наследование, абстракция данных, инкапсуляция, полиморфизм, достоинства ООП. Интерфейс, определение интерфейса, особенности интерфейсов, реализация интерфейсов, доступ к реализациям через ссылки, частичные реализации, отличия интерфейса от абстрактного класса, пример применения интерфейсов, переменные в интерфейсах, расширение интерфейсов. Пространство имен, пакеты, простые и составные имена, элементы и имена пакетов, имена классов и интерфейсов, определение пакета, поиск пакета и переменная среда CLASSPATH, защита доступа, категории видимости членов класса, доступ к членам класса, импорт пакетов, пакет java.lang.*, конфликты имен.

Тема 6. Обработка исключений

Исключение, исключение Java, общая форма обработки исключений, самостоятельная обработка исключений, множественный оператор catch, встроенные исключения Java.

Тема 7. Стандарт документирования Javadoc

Область применения комментариев, дескрипторы Javadoc и их назначение, примеры использования, генерация документации.

Тема 8. Коллекции в Java

Статические и динамические данные, динамические массива, списки, операции со списком, создание узла, добавление узла после заданного, стек, операции со стеком, задача с проверкой расстановки скобок, реализация стека (массив), инфиксная, префиксная, постфиксная форма записи арифметических выражений, вычисление арифметических выражений, очередь, реализация очереди (массив), дек. Коллекция, абстрактный класс AbstractCollection, особенности класса AbstractCollection, методы AbstractCollection, каркас коллекций, AbstractList, ArrayList, LinkedList, Stack, AbstactQueue, ограничение типа сохраняемых объектов.

4.3.1. Перечень семинарских, практических занятий и лабораторных работ

№ п/н	№ раздела и темы	Наименование семинаров, практических и лабораторных работ	Трудоемкость (час.)		Оценочные средства	Формируемые компетенции (индикаторы)*
			Всего часов	Из них практическая подготовка		
1	2	3	4	5	6	7
2	1	Тема 1. Класс StdDraw	2	2	Лабораторная работа	ОПК-2, ОПК-3
2	2	Тема 2. Массивы	8	2	Лабораторная работа	ОПК-2, ОПК-3
2	3	Тема 3. Символьный тип данных. Строки	3	2	Лабораторная работа	ОПК-2, ОПК-3
2	4	Тема 4. Рекурсия	3	4	Лабораторная работа	ОПК-2, ОПК-3
2	5	Тема 5. Объектно-ориентированное программирование	14	8	Лабораторная работа	ОПК-2, ОПК-3
2	6	Тема 6. Обработка исключений	1	2	Лабораторная работа	ОПК-2, ОПК-3
2	7	Тема 7. Стандарт документирования Javadoc	1	1	Лабораторная работа	ОПК-2, ОПК-3
2	8	Тема 8. Коллекции в Java	4	6	Лабораторная работа	ОПК-2, ОПК-3

4.3.2. Перечень тем (вопросов), выносимых на самостоятельное изучение студентами в рамках самостоятельной работы (СР)

Не предусмотрено

4.4. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОРГАНИЗАЦИИ САМОСТОЯТЕЛЬНОЙ РАБОТЫ СТУДЕНТОВ

Во время изучения дисциплины студент посещает лекции, практические занятия, выполняет лабораторные задания, готовится к тестам, зачетам и экзаменам. Для каждого вида деятельности необходимо правильно организовать самостоятельную работу.

Лекции. В высшем учебном заведении лекция является важной формой учебного процесса. На лекции студенты получают глубокие и разносторонние знания. Лекция способствует развитию творческих способностей, формирует идейную убежденность, позволяет устанавливать связь учебного материала с производством, новейшими научными достижениями. Лекция требует три вида деятельности: подготовку к лекции, работу на лекции и работу после лекции.

После прослушивания лекции студент должен проработать и осмыслить полученный материал. На каждый пример, приведенный на лекции, желательно, (если это возможно) привести свой. Материал, изложенный в лекции, можно просмотреть в других источниках.

В процессе лекционного занятия студент должен выделять важные моменты, выводы, анализировать основные положения. Недостаточно только «слушать» лекцию. Возможности памяти человека не универсальны. Как бы внимательно студент ни слушал лекцию, большая часть информации вскоре после восприятия будет забыта. Чтобы восстановить лекционный материал, его нужно повторить, а для этого лекцию необходимо конспектировать. Конспект лекций должен быть в отдельной тетради, в которой не должно быть ничего, кроме лекции. Не надо стремиться подробно слово в слово записывать всю лекцию. Конспектируйте только самое важное в рассматриваемой теме: ключевые слова и их значения, примеры использования конструкций, что старается выделить лектор, на чем акцентирует внимание студентов.

Тетрадь для конспекта лекций также требует особого внимания. Ее нужно сделать удобной, практичной и полезной, ведь именно она является основным информативным источником при подготовке к различным отчетным занятиям, зачетам, экзаменам. Конечно, оформление лекционной тетради – это дело вкуса. Но целесообразно отделить поля, где студент мог бы изложить свои мысли, вопросы, появившиеся в ходе лекции. Полезно одну из страниц оставлять свободной. Она потребуется потом, при самостоятельной подготовке. Сюда можно будет занести дополнительную информацию по данной теме, полученную из других источников: рисунки, схемы, примеры кода и т.д.

Лабораторное занятие. Лабораторные занятия по решению задач существенно дополняют лекции. В процессе анализа и решения задач студенты расширяют и углубляют знания, полученные из лекционного курса и учебников, приобретают умение применять общие закономерности к конкретным случаям.

Необходимо, чтобы студенты готовили теоретический материал, т.к. именно невыполнение этого требования приводит к неудаче при решении задач.

Несмотря на различие в видах задач, их решение можно проводить по следующему общему плану (некоторые пункты плана могут выпадать в некоторых конкретных случаях): а) прочесть внимательно условие задачи; б) посмотреть, все ли термины в условиях задачи известны и понятны (если что-то неясно, следует обратиться к учебнику, просмотреть решения предыдущих задач, посоветоваться с преподавателем); в) произвести анализ задачи, (нужно чётко понимать, в чем будет заключаться решение задачи); г) решить задачу; д) протестировать полученные решение на данных из примеров к задаче, а также на дополнительных данных.

Если задача не решена или «не решается», то необходимо еще раз вернуться к пунктам а) и б). Сколько раз нужно возвращаться к этим пунктам? Практика показывает, что не более десяти раз. Если и после этого задача «не решается», то можно попытаться найти решение этой или похожей задачи в различных источниках.

Домашнее задание. При выполнении домашнего задания необходимо просмотреть текст лекции, разобраться с новыми определениями, посмотреть задания, которые были выполнены на лабораторной работе и применить полученные знания для выполнения домашней работы.

Тест. В первую очередь постарайтесь узнать чего ждать от теста, какие примерно там будут задания. Если вам доступны образцы теста (как, например, при сдаче ЕГЭ), необходимо этим воспользоваться и ежедневно тренироваться.

Не оставляйте все на самый последний момент. Если будете постоянно готовиться к тесту, вы наверняка улучшите свои знания. Для этого составьте план на каждый день, чтобы правильно распределять свое время.

Делайте небольшие перерывы во время учебы. В промежутках можно дать себе небольшую физическую нагрузку. Мозг лучше всего работает, когда умственный труд сменяется физическим. Прогуляйтесь, побегайте, поиграйте в баскетбол, попинайте мяч – помимо стимуляции умственной деятельности, это снимет стресс.

Отдых и контроль над волнением — одни из главных составляющих успеха при подготовке к тесту. Часто ошибки совершаются только из-за стресса, который мешает сконцентрироваться и собраться. Чтобы быть отдохнувшим и расслабленным, соблюдайте составленный режим и старайтесь высыпаться.

Экзамен. На экзамене оцениваются: 1) понимание и степень усвоения теории; 2) методическая подготовка; 3) знание фактического материала; 4) знакомство с обязательной литературой; 5) умение приложить теорию к практике, решать практические задачи и т. д.; 6) логика, структура и стиль ответа, умение защищать выдвигаемые положения. Но значение экзаменов не ограничивается проверкой знаний. Являясь естественным завершением работы студента, они способствуют обобщению и закреплению знаний и умений, приведению их в строгую систему, а также устранению возникших в процессе занятий пробелов.

Студенты готовятся к экзаменам по-разному. Одни из них прорабатывают лишь некоторые вопросы, выбранные наугад, другие стремятся запомнить весь материал подряд, не вникая глубоко в его суть. Работа при этом концентрируется на одном стремлении – сдать экзамен. Недостатки такой системы очевидны. Очевидно также, что подготовка не должна ограничиваться чтением лекционных записей. Первоначальные необработанные конспекты студента содержат факты, определения, выводы, сделанные преподавателем, но в них, как правило, слабо просматривается связующая идея курса, так как студент, записывая каждую лекцию в отдельности, редко способен сразу и достаточно точно уловить общую направляющую мысль. Поэтому конспект требует дополнительной обработки на основе использования учебников и рекомендованной литературы.

Существенные недостатки имеет и такой способ подготовки к экзаменам, как беглый просмотр всего материала. Он эффективен только на некоторых этапах планирования и закрепляющего повторения. Более надежный и целесообразный путь – это тщательная систематизация материала при вдумчивом повторении, установлении внутрипредметных связей, увязке различных тем и разделов, закреплении путем решения задач.

Перед экзаменом назначается консультация. Цель ее – дать ответы на вопросы, возникшие в ходе самостоятельной подготовки. Хотелось бы обратить особое внимание на важность предэкзаменационных консультаций. Здесь студент имеет полную возможность получить ответ на все неясные ему вопросы. А для этого он должен проработать до консультации весь курс. Кроме того, преподаватель будет отвечать на вопросы других студентов, что будет для вас повторением и закреплением знаний. И еще очень важное обстоятельство: лектор на консультации, как правило, обращает внимание на те разделы, по которым на предыдущих экзаменах ответы были неудовлетворительными, а также фиксирует внимание на наиболее трудных разделах курса. Некоторые студенты не приходят на консультации либо потому, что считают, что у них нет вопросов к лектору, либо полагают, что у них и так мало времени и лучше самому почитать материал по конспекту или в

учебнике. Это глубокое заблуждение. Никакая другая работа не сможет принести столь значительного эффекта накануне экзамена, как консультация преподавателя.

Подготовку к экзамену следует начинать с первого дня изучения дисциплины. Как правило, на лекциях подчеркиваются наиболее важные и трудные вопросы или разделы курса, требующие внимательного изучения и обдумывания. Нужно эти вопросы выделить и обязательно постараться разобраться в них, не дожидаясь экзамена, проработать их, готовясь к лабораторным занятиям, попробовать самостоятельно решить несколько типовых задач. И если, несмотря на это, часть материала осталась неусвоенной, ни в коем случае нельзя успокаиваться, надеясь на то, что это не попадется на экзамене. Факты говорят об обратном: если те или другие вопросы курса не вошли в экзаменационный билет, преподаватель может их задать (и часто задает) в виде дополнительных вопросов. Точно такое же отношение должно быть выработано к вопросам и задачам, перечисленным в экзаменационной программе, выдаваемой студентам еще до экзамена. Обычно эти же вопросы и аналогичные задачи содержатся в экзаменационных билетах. Не следует оставлять без внимания ни одного раздела курса; если не удалось в чем-то разобраться самому, нужно обратиться к товарищам; если и это не помогло выяснить какой-либо вопрос до конца, нужно обязательно задать этот вопрос преподавателю на предэкзаменационной консультации. Чрезвычайно важно приучить себя к умению самостоятельно мыслить, учиться думать, понимать суть дела. Очень полезно после проработки каждого раздела восстановить в памяти содержание изученного материала, кратко записав это на листе бумаги. Если этого не сделать, то большая часть материала останется не понятой, а лишь формально заученной, и при первом же вопросе экзаменатора студент убедится в том, насколько поверхностно он усвоил материал.

4.5. ПРИМЕРНАЯ ТЕМАТИКА КУРСОВЫХ РАБОТ (ПРОЕКТОВ)

Не предусмотрено.

5. УЧЕБНО-МЕТОДИЧЕСКОЕ И ИНФОРМАЦИОННОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

а) перечень литературы

Основная

1. Баженова И. Ю. Языки программирования : учебник для студ. учреждений высш.проф. образования – Издат. центр "Академия", 2012. – 368 с. – ISBN: 978-5-7695-6856-5. – (ЭБС «Библиотех»).+
2. Программирование на языке Java: учеб. пособие для студентов направления «Информатика и вычислительная техника» / Э. А. Акчурин. – Самара : Изд-во ПГУТИ, 2011. – 317 с. – Режим доступа: ЭБС "РУКОНТ".+

Дополнительная

1. Парфилова Н. И., Пруцков А. В., Пылькин А. Н., Трусов Б. Г. Информатика и программирование. Основы информатики : учебник для студ. учреждений высш.проф. образования. – Издат. центр "Академия", 2012. – ISBN: 978-5-7695-4144-1. –256 с. – (ЭБС «БиблиоТех»).+

б) периодические издания

в) список авторских методических разработок:

Материалы курса, опубликованные в ИОС «DOMIC».

г) базы данных, информационно-справочные и поисковые системы

JDK 11 Documentation. URL: https://docs.oracle.com/en/java/javase/11/

6. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

6.1. УЧЕБНО-ЛАБОРАТОРНОЕ ОБОРУДОВАНИЕ:

Для проведения лекционных занятий необходима аудитория с презентационным оборудованием, для проведения лабораторных занятий необходима аудитория на 25-30 рабочих мест (в зависимости от численности учебной группы), оборудованная доской, презентационной техникой, компьютерами.

6.2. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ:

1. Комплект разработчика приложений Java Platform (JDK) 11, Standard Edition (распространяется бесплатно);

2. Интегрированная среда разработки NetBeans IDE 12 (распространяется бесплатно, LGPLv2.1, GPLv2 with Classpatch exception).

6.3. ТЕХНИЧЕСКИЕ И ЭЛЕКТРОННЫЕ СРЕДСТВА:

ИОС EDUCA, DOMIC, презентационное оборудование, персональный компьютер с возможностью демонстрации презентаций в формате ppt.

7. ОБРАЗОВАТЕЛЬНЫЕ ТЕХНОЛОГИИ

При реализации данного курса используются следующие образовательные технологии: технологии традиционного обучения, технологии проблемного обучения, технологии контекстного обучения, интерактивные технологии, технологии дистанционного обучения.

8. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

8.1. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ДЛЯ ВХОДНОГО КОНТРОЛЯ

Не предусмотрены.

8.2. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ТЕКУЩЕГО КОНТРОЛЯ

Разноуровневые задания к лабораторному практикуму. Каждое задание оценивается по 100-бальной шкале. Баллы за каждое задание представлены в таблице.

Учебная единица	Балл
java_stddraw / Класс StdDraw. Задача 2. График функции (Задания для самостоятельного выполнения)	2
java_stddraw / Класс StdDraw. Задача 1 (Задания для самостоятельного выполнения)	1
java_stddraw / Класс StdDraw. Задача 3. Полярные координаты (Задания для самостоятельного выполнения)	1
java_array1 / Одномерные массивы. Задача 1 (Задания для самостоятельного выполнения)	0.5
java_array1 / Одномерные массивы. Задача 2 (Задания для самостоятельного выполнения)	0.5
java_array1 / Одномерные массивы. Задача 3 (Задания для самостоятельного выполнения)	0.5
java_array1 / Одномерные массивы. Задача 4 (Задания для самостоятельного выполнения)	0.5
java_array1 / Одномерные массивы. Задача 5 (Задания для самостоятельного выполнения)	0.5
java_array1 / Одномерные массивы. Задача 6 (Задания для самостоятельного выполнения)	0.5
java_array1 / Одномерные массивы. Задача 7 (Задания для самостоятельного выполнения)	0.5
java_array1 / Одномерные массивы. Задача 8 (Задания для самостоятельного выполнения)	0.5
java_array1 / Одномерные массивы. Задача 9. Сортировка и поиск (Задания для самостоятельного выполнения)	1
java_array1 / Одномерные массивы. Задача 10 (Задания для самостоятельного выполнения)	0.5
java_array1 / Одномерные массивы. Задача 11 (Задания для самостоятельного выполнения)	1
java_array2 / Многомерные массивы. Задача 1 (Задания для самостоятельного выполнения)	2
java_array2 / Многомерные массивы. Задача 2 (Задания для самостоятельного выполнения)	1
java_array2 / Многомерные массивы. Задача 3 (Задания для самостоятельного выполнения)	2
java_array2 / Многомерные массивы. Задача 4 (Задания для самостоятельного выполнения)	1
java_array2 / Многомерные массивы. Задача 5. Случайное блуждание (Задания для самостоятельного выполнения)	2.5
java_string / Символьный тип данных и строки. Задача 1 (Задания для самостоятельного выполнения)	0.5
java_string / Символьный тип данных и строки. Задача 2 (Задания для самостоятельного выполнения)	0.5
java_string / Символьный тип данных и строки. Задача 3 (Задания для самостоятельного выполнения)	0.5
java_string / Символьный тип данных и строки. Задача 4 (Задания для самостоятельного	0.5

выполнения)	
java_string / Символьный тип данных и строки. Задача 5 (Задания для самостоятельного выполнения)	0.5
java_string / Символьный тип данных и строки. Задача 6 (Задания для самостоятельного выполнения)	0.5
java_string / Символьный тип данных и строки. Задача 7 (Задания для самостоятельного выполнения)	0.5
java_string / Символьный тип данных и строки. Задача 8 (Задания для самостоятельного выполнения)	1
java_string / Символьный тип данных и строки. Задача 13 (Задания для самостоятельного выполнения)	1
java_string / Символьный тип данных и строки. Задача 14 (Задания для самостоятельного выполнения)	1
java_string / Символьный тип данных и строки. Задача 15 (Задания для самостоятельного выполнения)	1
java_recursion / Рекурсия. Задача 0 (Задания для самостоятельного выполнения)	1
java_recursion / Рекурсия. Задача 1 (Задания для самостоятельного выполнения)	0.5
java_recursion / Рекурсия. Задача 2 (Задания для самостоятельного выполнения)	1
java_recursion / Рекурсия. Задача 3 (Задания для самостоятельного выполнения)	1
java_recursion / Рекурсия. Задача 4. Файловая система (Задания для самостоятельного выполнения)	2.5
java_recursion / Рекурсия. Задача 5. Рекурсивная графика (Задания для самостоятельного выполнения)	2.5
java_oop / ООП-1. Точка (Задания для самостоятельного выполнения)	2
java_oop / ООП-2. Массив объектов (Задания для самостоятельного выполнения)	2
java_oop / ООП-3. Черепашья графика (Задания для самостоятельного выполнения)	3
java_oop / ООП-4. Чтение объектов из файла (Задания для самостоятельного выполнения)	3
java_oop / ООП-6. Геометрические фигуры (Задания для самостоятельного выполнения)	4.5
java_oop / ООП-7. Последовательности (Задания для самостоятельного выполнения)	2
java_exception / Обработка исключений. Задача 1 (Задания для самостоятельного выполнения)	1
java_exception / Обработка исключений и JavaDoc. Геометрические фигуры - 2 (Задания для самостоятельного выполнения)	2
java_cf / Коллекции - 1. Задача со скобками (Задания для самостоятельного выполнения)	2
java_cf / Коллекции - 2. Стек. Вычисление выражений (Задания для самостоятельного выполнения)	2
java_cf / Коллекции - 3. Сортировка списков (Задания для самостоятельного выполнения)	1.5
java_cf / Коллекции - 4. Множества (Задания для самостоятельного выполнения)	2
java_cf / Коллекции - 5. Построение индекса (Задания для самостоятельного выполнения)	3

Примеры оценочных средств текущего контроля

1. Лабораторная работа по теме «Класс StdDraw»

В варианте дана функция вида $y = f(x)$.

Напишите программу построения графика функции из вашего варианта на заданном отрезке $[min, max]$ (значение границ отрезка вводит пользователь с консоли). График должен содержать оси. Единицы масштаба по осям X и Y должны совпадать (для контроля необходимо вывести график функции $y = x$. Все поле рисования должно быть разграфлено как тетрадь в клетку (выберите нейтральный цвет, например светло-зеленый).

При решении задачи используйте процедурную декомпозицию.

Рекомендация. В случае получения в качестве значения функции положительной или отрицательной бесконечности или NaN, не нужно выводить соответствующую точку на графике функции.
для проверки вам потребуются статические методы `boolean Double.isInfinite()` и `boolean Double.isNaN(double)`.

Дополнительное задание. Измените программу таким образом, чтобы она допускала построение графика функции с другими заданными коэффициентами (например, если в качестве задания дана функция $y = \sin(x)$, то программа должна допускать построение графика функции $y = a \cdot \sin(b \cdot x + c) + d$ (значение коэффициентов вводит пользователь с консоли).

2. Лабораторная работа по теме «Коллекции в Java»

1. На основе класса `java.util.HashSet` создайте множество объектов типа `Point3d` (исходный код класса перенесите в текущий проект).
2. Выполните 1000 итераций добавления в созданную коллекцию случайных точек с целочисленными координатами в пределах от 0 до 4.
3. Используя метод `size()`, выведите размерность множества, а с помощью метода `contains` проверьте содержит ли полученное множество точку с координатами (0,0,0).

Внимание! Для корректной работы метода необходимо в классе `Point3d` реализовать методы `equals()` и `hashCode()` в соответствии с соглашением Java.

4. Реализуйте и продемонстритуйте метод согласно вашему варианту, для решения задачи используйте интерфейс `Iterator` и соответствующие методы.
5. Выведите полученную коллекцию и количество ее элементов в консоль.
6. Используя класс `java.util.TreeSet` создайте множество объектов исходной коллекции (для этого вам понадобится конструктор `TreeSet(Collection c)`)

Внимание! Для работы с коллекцией `TreeSet` необходимо в классе `Point3d` реализовать интерфейс `Comparable`, установите порядок по расстоянию (евклидовому или «манхэттенскому» на ваш выбор) от точки начала координат.

7. Выведите полученную коллекцию и количество ее элементов в консоль.

3. Вопросы для собеседования по теме «Объектно-ориентированное программирование»

1. Зачем нужна абстракция данных? Зачем создавать классы?
2. Зачем делать различие между примитивными и ссылочными типами данных? Почему не оставить только ссылочные?
3. У каждого ли класса должен быть конструктор?
4. Что такое null?
5. Что случится, если не написать ключевое слово new при создании экземпляра класса?
6. Почему нет смысла печатать объекты с помощью конструкции System.out.print(obj.toString())?
7. Если в классе не переопределить метод toString() что произойдет при попытке вывода на экран объекта этого класса?
8. Может ли класс, реализующий тип данных, содержать статический метод?
9. Что произойдет, если обратиться к аргументу или методу с модификатором доступа private из другого класса?
10. Как из конструктора класса наследника обратиться к конструктору класса родителя?
11. Можно ли в ссылочную переменную подкласса записать объект, созданный от суперкласса?
12. Что такое переопределение методов? Отличается ли это понятие от перегрузки методов?
13. Как запретить переопределение методов в подклассах?
14. Что такое абстрактный метод? Для чего он нужен? В каких классах его можно объявить?
15. Можно ли создать абстрактный класс без абстрактных методов?
16. Можно ли объявить класс abstract и final одновременно?
17. Может ли класс наследовать 2 и более класса?
18. Может ли подкласс быть абстрактным, если его супер-класс не является таковым?
19. Пусть A – интерфейс, можно ли создать объект следующим образом new A()?
20. Пусть A – интерфейс, можно ли создать ссылочную переменную следующим образом A x? На какие объекты она может ссылаться?

8.3. ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПРОМЕЖУТОЧНОГО КОНТРОЛЯ

- баллы за работу в семестре (выполнение лабораторных работ) – 65 баллов
- баллы за экзаменационный тест – 30 баллов
- баллы за посещаемость – 5 баллов

Критерии итоговой оценки за курс:

- 80 и более – отлично
- от 70 до 79 – хорошо
- от 60 до 69 и – удовлетворительно
- менее 60 – неудовлетворительно

В случае, если студент не набрал необходимое количество баллов, проводится собеседование, на котором ему дается дополнительная задача и вопросы из списка. Ответ на каждый вопрос и задачу оценивается из 5 баллов. Студенту случайным образом дается нужное количество вопросов, но не боле 4.

Если студент в течение семестра набрал менее 40 баллов, тогда на промежуточной аттестации ему выставляется неудовлетворительно.

Список вопросов для экзаменационного теста и собеседования

1. Соглашения по оформлению кода на Java
2. Одномерные массивы, объявление, особенности, ввод с клавиатуры, поэлементные операции, максимальный элемент массива.
3. Сортировка массивов. Метод пузырька. Метод пузырька с флажком. Реализация алгоритма.
4. Сортировка массивов. Метод выбора. Реализация алгоритма.
5. Поиск в массиве, линейный поиск, двоичный поиск. Реализация алгоритма двоичного поиска. Сравнение методов.
6. Методы класса Arrays. Примеры.
7. Многомерные массивы, объявление, ввод и вывод, неоднородные (нерегулярные) массивы, инициализация многомерных массивов, получение размерности массива. Примеры.
8. Кодирование символов. Одно- и двухбайтовые кодировки. Символьный тип данных и его особенности. Специальные символы.
9. Класс String и его методы. Ввод строк с клавиатуры. Примеры.
10. Эквивалентность строк. Сравнение строк.
11. Изменяемые и неизменяемые строки. Класс StringBuffer. Примеры.
12. Рекурсия. Примеры рекурсивных объектов. Связь рекурсии и метода математической индукции.
13. Основное правило рекурсии. Рекурсия и итерация. Примеры. Особенности рекурсивных методов.
14. Стек вызовов. Переполнение стека вызовов. Проблемы рекурсии, примеры.
15. Анализ вычисления последовательности чисел Фибоначчи с использованием рекурсивного метода.
16. Рекурсия. Вспомогательные методы. Рекурсивная сортировка методом выбора.
17. Объектно-ориентированное программирование. Понятие объекта и класса.
18. Использование классов. Общая форма класса. Члены класса. Объявление объектов. Присваивание переменным объектных ссылок. Примеры.
19. Методы классы, методы чтения и модификаторы. Конструкторы и их особенности.
20. Ключевое слово this. Сокрытие переменной экземпляра. Метод toString(). Примеры.
20. Способы передачи аргументов методу, их различия. Примеры. Ключевое слово static, ограничения. Ключевое слово final.

21. Аргументы переменной длины, особенности, перегрузка, неопределенность. Примеры.

22. Наследование. Понятие подкласса и суперкласса. Общая форма наследования. Доступ к членам суперкласса. Ссылочная переменная суперкласса.

23. Наследование. Ключевое слово super, вызов конструктора суперкласса. Обращение к членам суперкласса.

24. Наследование. Класс Object и его методы. Переопределение метода equals.

25. Наследование. Порядок вызова конструкторов. Конструктор по умолчанию. Затенение полей родительского класса. Переопределение методов. Динамическая диспечерезация методов.

26. Абстрактные методы. Абстрактные классы и их особенности.

27. Использование ключевого слова final при наследовании.

28. Проблема ромба (множественное наследование). Интерфейс и его особенности. Определение интерфейса. Реализация интерфейса в классе.

29. Доступ к реализациям через интерфейс. Частичные реализации интерфейса.

30. Отличие интерфейса от абстрактного класса. Переменные в интерфейсах. Расширение интерфейсов.

31. Доступность членов класса.

32. Принципы объектно-ориентированного программирования. Причины использования и следствия. Достоинства ООП.

33. Пакеты. Пространство имен. Простые и составные имена. Объявление пакета. Импорт пакетов.

34. Пакет java.lang. Конфликты имен.

35. Класс BigInteger. Константы и методы класса BigInteger.

36. Исключение. Обработка исключений. <<Опасные>> методы. Общая форма обработки исключений. Множественные операторы catch. Блок finally. Встроенные исключения Java.

37. Исключение. Операторы throw и throws. Иерархия исключений. Исключения и полиморфизм. Пользовательские классы исключений.

38. Оболочки примитивных типов. Класс Number. Классы Double и Float. Конструкторы, константы, методы. Автоупаковка и автораспаковка.

39. Оболочки примитивных типов. Класс Number. Классы Byte, Short, Integer и Long. Конструкторы, константы, методы. Автоупаковка и автораспаковка.

40. Преобразование типов. Расширение объектных типов. Сужение объектных типов. Оператор instanceof.

41. Структуры данных. Примеры структур данных и их свойства. Линейный список. Типы списков.

42. Структура данных стек. Реализация стека на основе массива.

43. Структура данных стек. Задача о правильной расстановке скобок (один вид скобок, несколько видов скобок).

44. Структура данных стек. Инфиксная, префиксная и постфиксная формы записи арифметических выражений. Алгоритм вычисления арифметических выражений в постфиксной форме.

45. Коллекции Java. Понятие коллекции. Интерфейсы коллекций. Модифицирующие и немодифицирующие операции коллекций. Итераторы. Методы итераторов.

46. Списки. Интерфейс List. Операции со списками. Итератор по списку и его методы.

Классы ArrayList, конструкторы, особенности. Класс LinkedList, конструкторы, методы, применение. Сравнение классов ArrayList и LinkedList.

47. Очередь. Интерфейс Queue. Методы очередей. Интерфейс Deque и класс ArrayDeque.

Множество. Интерфейс Set. Сравнение элементов. Операции над множествами. Классы HashSet, LinkedHashSet, TreeSet. Хеширование. Метод hashCode() и его согласованность с методом equals().

48. Отображение. Интерфейс Map. Методы отображений. Пары -- интерфейс Map.Entry. Классы HashMap, LinkedHashMap, TreeMap.

49. Сравнение элементов. Интерфейсы Comparable и Comparator их применение и сравнение. Упорядоченные множества и отображения.

50. Алгоритмы для коллекций. Класс Collections.

Примеры дополнительных задач для собеседования на промежуточной аттестации:

1. Пример вопроса по теме Интерфейсы

Даны три класса ClassA, ClassB и ClassC, а также два интерфейса InterfaceA и InterfaceB. какие из записей будут корректными?

Выберите один или несколько ответов:

- [ ]
```
class ClassA extends ClassB, ClassC {
    ...
}
```
- [ ]
```
class ClassA, ClassB extends ClassC implements InterfaceA {
    ...
}
```
- [ ]
```
class ClassA extends ClassB implements InterfaceA {
    ...
}
```
- [ ]
```
class ClassA, ClassB extends ClassC {
    ...
}
```
- [ ]
```
class ClassA extends ClassB {
    ...
}
```
- [ ]
```
class ClassA extends ClassB implements InterfaceA, InterfaceB {
    ...
}
```

Разработчики:

_____	_____доцент____	_____ Зинченко А.С._____
(подпись)	(занимаемая должность)	(Ф.И.О.)

Программа составлена в соответствии с требованиями ФГОС ВО по направлению подготовки 02.03.02 «Фундаментальная информатика и информационные технологии» (уровень бакалавриата), утвержденный приказом Министерства образования и науки Российской Федерации от 23 августа 2017 г. N 808, зарегистрированный в Минюсте России «14» сентября 2017 г. № 48185 с изменениями и дополнениями с изменениями и дополнениями от: 26 ноября 2020 г., 8 февраля 2021 г.