

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Астраханский государственный университет имени В. Н. Татищева»
(Астраханский государственный университет им. В. Н. Татищева)

СОГЛАСОВАНО
Руководитель ОПОП

_____ Ю.А. Головкин

«13» июня 2024 г.

УТВЕРЖДАЮ
Заведующий кафедрой ИТ

_____ А.Н. Марьенков

«13» июня 2024 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

«ТЕХНОЛОГИИ JAVA»

Составитель(и)

**Железняков Д.В., ассистент; Морозов Б.Б., к.х.н.,
доцент кафедры ИТ;**

Направление подготовки /
специальность

09.03.02 Информационные системы и технологии

Направленность (профиль) ОПОП

**Технологии разработки и администрирования
информационных систем**

Квалификация (степень)

бакалавр

Форма обучения

очно-заочная

Год приёма

2022

Курс

3

Семестр(ы)

5

Астрахань – 2024

1. ЦЕЛИ И ЗАДАЧИ ОСВОЕНИЯ ДИСЦИПЛИНЫ (МОДУЛЯ)

1.1. Целями освоения дисциплины (модуля) «Технологии Java» является изучение студентами современного подхода к программированию на основе объектно-ориентированной технологии, приобретение студентами навыков разработки программного обеспечения на языке Java.

1.2. Задачи освоения дисциплины (модуля):

- изучение основных принципов объектно-ориентированного подхода, и их реализация в языке Java, методов объектной реализации стандартных библиотек, способов организации и структуры объектно-ориентированных программ;
- приобретение студентами навыков разработки программного обеспечения на языке Java;
- разработка средств реализации информационных технологий (методические, информационные, математические, алгоритмические, технические и программные) с применением объектного подхода и объектно-ориентированных инструментальных средств; разработка и внедрение технологий разработки объектов профессиональной деятельности с применением платформы Java;
- инсталляция и отладка объектно-ориентированных программных средств для ввода информационных систем в эксплуатацию;
- сборка объектной программной системы из готовых компонентов – классов и объектов;
- составление инструкций по эксплуатации объектно-ориентированных информационных систем.

2. МЕСТО ДИСЦИПЛИНЫ (МОДУЛЯ) В СТРУКТУРЕ ОПОП

2.1. Учебная дисциплина (модуль) «Технологии Java» относится к части, формируемой участниками образовательных отношений и осваивается в 5 семестре.

2.2. Для изучения данной учебной дисциплины (модуля) необходимы следующие знания, умения, навыки, формируемые предшествующими учебными дисциплинами (модулями):

- Информатика;
- Основы программирования.

Знания: основные понятия информатики, принципы действия технических средств информатики – компьютеров, программное обеспечение компьютеров, базовые типы данных.

Умения: работа в интегрированных системах программирования, применение базовых алгоритмов в программировании.

Навыки: разработки алгоритмов, применение инструментария интегрированных сред программирования для решения различных прикладных задач.

2.3. Последующие учебные дисциплины (модули) и (или) практики, для которых необходимы знания, умения, навыки, формируемые данной учебной дисциплиной (модулем):

- Инженерный практикум;
- Разработка мобильных приложений;
- Программная инженерия.

3. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

Процесс освоения дисциплины (модуля) направлен на формирование элементов

следующей(их) компетенции(ий) в соответствии с ФГОС ВО и ОПОП ВО по данному направлению подготовки / специальности:

а) профессиональной (ПК): ПК-2 Способен разрабатывать программное обеспечение, включая проектирование, отладку, проверку работоспособности и модификацию ПО.

Таблица 1 – Декомпозиция результатов обучения

Код и наименование компетенции	Планируемые результаты обучения по дисциплине (модулю)		
	Знать (1)	Уметь (2)	Владеть (3)
ПК–2 Способен разрабатывать программное обеспечение, включая проектирование, отладку, проверку работоспособности и модификацию ПО.	ИПК-2.1.1 современные информационные технологии разработки, отладки, проверки работоспособности, модификации программного обеспечения	ИПК-2.2.1 осуществлять выбор информационных технологий для решения задач по разработке, отладке, проверке работоспособности, модификации программного обеспечения	ИПК-2.3.1 навыками разработки, отладки, проверки работоспособности, модификации программного обеспечения с использованием современных информационных технологий

4. СТРУКТУРА И СОДЕРЖАНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

Объем дисциплины (модуля) составляет 5 зачётных(ые) единиц(ы), в том числе 36 часов(а), выделенных на контактную работу обучающихся с преподавателем (36 часов(а) – лабораторные работы), и 126 часов(а) – на самостоятельную работу обучающихся.

Таблица 2 – Структура и содержание дисциплины (модуля)

Раздел, тема дисциплины (модуля)	Семестр	Контактная работа (в часах)			Самост. работа		Форма текущего контроля успеваемости, форма промежуточной аттестации
		Л	ПЗ	ЛР	КР	СР	
Объектная модель. Классы и объекты	5	0	0	5	2	18	Отчет по лабораторной работе, вопросы к экзамену
Основы языка Java		0	0	5	2	18	Отчет по лабораторной работе, вопросы к экзамену
Механизм классов		0	0	5	2	18	Отчет по лабораторной работе, вопросы к экзамену
Наследование		0	0	5	2	18	Отчет по лабораторной работе, вопросы к экзамену

Раздел, тема дисциплины (модуля)	Семестр	Контактная работа (в часах)			Самост. работа		Форма текущего контроля успеваемости, форма промежуточной аттестации
		Л	ПЗ	ЛР	КР	СР	
Классы коллекций		0	0	5	2	18	Отчет по лабораторной работе, вопросы к экзамену
Обработка исключений		0	0	5	4	18	Отчет по лабораторной работе, вопросы к экзамену
Ввод-вывод и сериализация		0	0	6	4	18	Отчет по лабораторной работе, вопросы к экзамену
Итого		0	0	36	18	126	Экзамен

Примечание: Л – лекция; ПЗ – практическое занятие, семинар; ЛР – лабораторная работа; КР – курсовая работа; СР – самостоятельная работа.

Таблица 3 – Матрица соотношения разделов, тем учебной дисциплины (модуля) и формируемых компетенций

Раздел, тема дисциплины (модуля)	Кол-во часов	Код компетенции	Общее количество компетенций
		ПК-2	
Объектная модель. Классы и объекты	25	+	1
Основы языка Java	25	+	1
Механизм классов	25	+	1
Наследование	25	+	1
Классы коллекций	25	+	1
Обработка исключений	27	+	1
Ввод-вывод и сериализация	28	+	1
Итого	180		1

Краткое содержание каждой темы дисциплины (модуля)

Объектная модель. Классы и объекты.

Эволюция объектной модели. Составные части объектного подхода. Применение объектной модели. Природа объекта. Отношения между объектами. Природа классов. Отношения между классами. Взаимосвязь классов и объектов. Качество классов и объектов. Парадигмы ООП. Абстракция, инкапсуляция, наследование, полиморфизм.

Основы языка Java.

Происхождение Java. Базовые термины Java. Лексические элементы языка. Типы данных, переменные и массивы в Java. Целые типы. Типы с плавающей запятой. Символьный тип. Булевский тип. Переменные. Преобразование и приведение типов. Массивы. Операции Java. Арифметические операции. Поразрядные операции. Операции булевой логики. Условная операция. Приоритеты операций. Управляющие операторы Java. Операторы выбора. Операторы цикла. Операторы перехода.

Механизм классов.

Основы классов. Объявление классов. Модификаторы. Объявление объектов. Конструкторы. Перегрузка конструкторов. Цепочки конструкторов. Управление памятью в Java. Сборка мусора. Объявление полей. Объявление и использование методов. Перегрузка методов. Статические поля и методы. Управление доступом. Пакеты. Вложенные и внутренние классы.

Наследование.

Отношения между классами: ассоциация, наследование. Основы наследования. Наследование полей и методов. Использование суперклассов. Переопределение методов. Создание многоуровневой иерархии. Наследование и конструкторы. Объектный полиморфизм. Восходящее и нисходящее приведение классов. Динамическая диспетчеризация методов. Абстрактные классы. Интерфейсы. Объявление и реализация интерфейсов. Подинтерфейсы. Использование интерфейсов и абстрактных классов.

Классы коллекций.

Типы коллекций. Абстрактные типы данных. Основные интерфейсы коллекций: список, множество и словарь. Базовые операции над коллекциями. Нетипизированные и типизированные коллекции. Сравнение объектов. Интерфейсы Comparable и Comparator. Сортированные коллекции. Итераторы. Реализации коллекций, производительность реализаций. Алгоритмы обработки коллекций.

Обработка исключений.

Понятие об исключениях. Типы исключений. Иерархия классов исключений. Использование операторов try, catch, throws и finally. Обработка вложенных исключений. Встроенные исключения Java. Создание собственных классов исключений.

Ввод-вывод и сериализация.

Потоки ввода-вывода. Типы потоков, байтовые и символьные потоки. Создание потоков. «Цепочки» потоков. Фильтрующие потоки. Консольный ввод-вывод. Работа с файловой системой. Чтение и запись файлов. Сериализация и десериализация. Интерфейс сериализации. Версии классов. Сетевые возможности Java. Порты. Работа с протоколом HTTP. Сокеты. Использование сокетов.

5. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ПРЕПОДАВАНИЮ И ОСВОЕНИЮ ДИСЦИПЛИНЫ (МОДУЛЯ)

5.1. Указания для преподавателей по организации и проведению учебных занятий по дисциплине (модулю)

Учебная деятельность студента в процессе изучения строится из контактных форм работы с преподавателем (аудиторные занятия, экзамен) и самостоятельной работы.

Для успешного освоения дисциплины является обязательным посещение всех занятий, выполнение домашнего задания и лабораторно-практических работы, которые назначаются преподавателем.

Методическая поддержка дисциплины обеспечивается использованием дистанционных технологий. Студентам предлагается информационный ресурс, расположенный по адресу: <http://moodle.asu.edu.ru>, на сервере дистанционного обучения АГУ. Доступ студентов к учебным ресурсам осуществляется по учетной записи и паролю после регистрации на курс «Технологии Java» на период обучения по данной дисциплине. На сервере размещен методический материал по данной дисциплине, в содержание которого входит: теоретический материал и указания по выполнению лабораторно-практических работ; вопросы к экзамену.

Аудиторные занятия проводятся на основе теоретического материала, опубликованного на образовательном портале, это позволяет студентам изучить пропущенный материал или самостоятельно разобраться с темой, не освоенной на занятии.

Для исключения отрыва студентов от учебного процесса проводится учет посещаемости аудиторных занятий.

5.2. Указания для обучающихся по освоению дисциплины (модулю)

Для освоения дисциплины «Технологии Java» студентам рекомендуется организация самостоятельной работы по следующим видам работ:

- самостоятельное изучение теоретического материала дисциплины с использованием Internet-ресурсов, информационных баз, методических разработок, специальной учебной и научной литературы;
- дополнительная подготовка к лабораторно-практическим работам или выполнение части лабораторной работы, которую не успели сделать в аудитории;
- подготовка к экзамену.

Теоретический материал, задания к лабораторно-практическим занятиям размещены на образовательном портале <http://moodle.asu.edu.ru>.

В процессе подготовки к аудиторным занятиям, необходимо обратить особое внимание на самостоятельное изучение рекомендованной литературы. Самостоятельная работа с 6 учебниками, учебными пособиями, научной, справочной литературой, материалами периодических изданий и Интернета является наиболее эффективным методом получения дополнительных знаний, позволяет значительно активизировать процесс овладения информацией, способствует более глубокому усвоению изучаемого материала.

Таблица 4 – Содержание самостоятельной работы обучающихся

Вопросы, выносимые на самостоятельное изучение	Кол-во часов	Форма работы
Объектная модель. Классы и объекты	18	отчет о выполнении ЛР, Устный опрос на экзамене
Основы языка Java	18	отчет о выполнении ЛР, Устный опрос на экзамене
Механизм классов	18	отчет о выполнении ЛР, Устный опрос на экзамене
Наследование	18	отчет о выполнении ЛР, Устный опрос на экзамене
Классы коллекций	18	отчет о выполнении ЛР, Устный опрос на экзамене
Обработка исключений	18	отчет о выполнении ЛР, Устный опрос на экзамене
Ввод-вывод и сериализация	18	отчет о выполнении ЛР, Устный опрос на экзамене

5.3. Виды и формы письменных работ, предусмотренных при освоении дисциплины (модуля), выполняемые обучающимися самостоятельно

Письменные работы, предусмотренные при освоении дисциплины, выполняемые студентами самостоятельно – это отчет по выполнению лабораторных практических работ. Тематика ЛПР представлена в таблице 4. Все отчеты по ЛПР оформляются в виде программного кода на языке программирования JAVA.

Отчеты по проектной работе оформляются в виде программного кода на языке программирования Python, а также отчет в программе WORD согласно требованию ГОСТа 7.32- 2017 «Отчет о научно-исследовательской работе» о подготовке, содержанию и оформлению научно-исследовательских и курсовых работ.

Требования к оформлению и представлению отчета по курсовой работе

Курсовая работа оформляется в виде программного кода на языке программирования Python. Отчет должен отвечать общим требованиям, предъявляемым к научно-

исследовательской работе и другой проектной документации, поэтому структура, требования к содержанию и оформлению отчета и иллюстрационного альбома должны соответствовать ГОСТ 7.32-2017 «Отчет о научно-исследовательской работе» а графического материала – Единой системе конструкторской документации (ГОСТ 2.104-68, ГОСТ 2.301-68 и др.). Правила оформления схем алгоритмов и программных продуктов по ГОСТ 19.002-80.

Отчет является основным отчетным документом, который содержит систематизированные данные о выполненной студентом работе, решений, иллюстрации, схемы, графики.

Общим требованием к отчету являются: четкость и логическая последовательность изложения материала, убедительность аргументации, краткость и ясность формулировок, исключающих неоднозначность толкования, конкретность изложения результатов, доказательств и выводов.

Отчет выполняется в приложении MSWord.

Отчет должен включать следующие структурные элементы, располагающиеся в строгой последовательности:

ТИТУЛЬНАЯ ЧАСТЬ:

Титульный лист (первый лист документа);

Оглавление;

ЗАДАНИЕ;

ВВЕДЕНИЕ;

ОСНОВНАЯ ЧАСТЬ

ЗАКЛЮЧЕНИЕ (выводы)

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЯ (программная документация, схемы, результаты моделирования, таблицы, графики и т.п.).

Объем отчета не должен превышать 35 стр. Объем и состав демонстрационных материалов определяется требованиями индивидуального задания.

6. ОБРАЗОВАТЕЛЬНЫЕ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

В рамках реализации компетентностного подхода в соответствии с требованиями ФГОС ВО в учебном процессе предусмотрены активные и интерактивные формы проведения занятий.

Основой для выстраивания аудиторных занятий является лабораторные работы. Это самостоятельная работа учащегося, выполненная с помощью консультаций преподавателя. Основное отличие такой деятельности — это то, что студент, прежде всего, получают практические навыки в области программирования.

6.1. Образовательные технологии

Для достижения поставленных целей преподавания дисциплины реализуются следующие образовательные технологии:

- самостоятельное изучение теоретического материала дисциплины с использованием Internet-ресурсов, информационных баз, методических разработок, специальной учебной и научной литературы;

- закрепление теоретического материала при проведении лабораторных работ, выполнения проблемно-ориентированных, поисковых, творческих заданий.

Цели дисциплины достигаются путем сочетания комплекса методов обучения: проведения лабораторно-практических занятий на ЭВМ и организации самостоятельной работы студентов. Лабораторные работы ориентированы на формирование деятельностных компетентностей. Они заключаются в выполнении сквозного цикла лабораторных работ. В процессе выполнения лабораторных работ достигаются следующие цели:

- изучаются инструментальные средства программных продуктов;
- формируются практические навыки обработки информации различного вида и формы при решении конкретных практических задач;
- формируется навык выявления ошибочных и нестандартных ситуаций и реагирования на них.

Таблица 5 – Образовательные технологии, используемые при реализации учебных занятий

Раздел, тема дисциплины (модуля)	Форма учебного занятия		
	Лекция	Практическое занятие, семинар	Лабораторная работа
Объектная модель. Классы и объекты	Не предусмотрено	Не предусмотрено	выполнение лабораторной работы
Основы языка Java	Не предусмотрено	Не предусмотрено	выполнение лабораторной работы
Механизм классов	Не предусмотрено	Не предусмотрено	выполнение лабораторной работы
Наследование	Не предусмотрено	Не предусмотрено	выполнение лабораторной работы
Классы коллекций	Не предусмотрено	Не предусмотрено	выполнение лабораторной работы
Обработка исключений	Не предусмотрено	Не предусмотрено	выполнение лабораторной работы
Ввод-вывод и сериализация	Не предусмотрено	Не предусмотрено	выполнение лабораторной работы

Учебные занятия по дисциплине могут проводиться с применением информационно-телекоммуникационных сетей при опосредованном (на расстоянии) интерактивном взаимодействии обучающихся и преподавателя в режимах on-line в формах: видеолекций, лекций-презентаций, видеоконференции, собеседования в режиме чат, форума, чата, выполнения виртуальных практических и/или лабораторных работ и др. Максимальный объем занятий обучающегося с применением электронных образовательных технологий не должен превышать 25%

6.2. Информационные технологии

- использование возможностей интернета в учебном процессе (использование сайта преподавателя (рассылка заданий, предоставление выполненных работ, ответы на вопросы, ознакомление обучающихся с оценками и т. д.));
- использование электронных учебников и различных сайтов (например, электронных библиотек, журналов и т. д.) как источников информации;
- использование возможностей электронной почты преподавателя;
- использование средств представления учебной информации (электронных учебных пособий и практикумов, применение новых технологий для проведения очных (традиционных) лекций и семинаров с использованием презентаций и т. д.);

- использование интегрированных образовательных сред, где главной составляющей являются не только применяемые технологии, но и содержательная часть, т. е. информационные ресурсы (доступ к мировым информационным ресурсам, на базе которых строится учебный процесс);
- использование виртуальной обучающей среды (LMS Moodle «Электронное образование») и

6.3. Программное обеспечение, современные профессиональные базы данных и информационные справочные системы

6.3.1. Программное обеспечение

– Лицензионное программное обеспечение

Adobe Reader – Программа для просмотра электронных документов

Платформа дистанционного обучения LMS Moodle – Виртуальная обучающая среда

Mozilla FireFox – Браузер

Microsoft Office 2013, Microsoft Office Project 2013, Microsoft Office Visio 2013 – Офисная программа

7-zip – Архиватор

Microsoft Windows 10 Professional – Операционная система

Kaspersky Endpoint Security – Средство антивирусной защиты

Eclipse – Среда разработки

6.3.2. Современные профессиональные базы данных и информационные справочные системы

1. Электронный каталог Научной библиотеки АГУ на базе MARKSQL НПО «Информ-систем». <https://library.asu.edu.ru>.
2. Электронный каталог «Научные журналы АГУ». <http://journal.asu.edu.ru>.
3. Универсальная справочно-информационная база данных периодических изданий ООО "ИВИС". <http://dlib.eastview.com>.
4. Корпоративный проект Ассоциации региональных библиотечных консорциумов (АРБИКОН) «Межрегиональная аналитическая роспись статей» (МАРС) - сводная база данных, содержащая полную аналитическую роспись 1800 названий журналов по разным отраслям знаний. Участники проекта предоставляют друг другу электронные копии отсканированных статей из книг, сборников, журналов, содержащихся в фондах их библиотек. <http://mars.arbicon.ru>.

7. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ТЕКУЩЕГО КОНТРОЛЯ И ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

7.1. Паспорт фонда оценочных средств

При проведении текущего контроля и промежуточной аттестации по дисциплине (модулю) «Технологии Java» проверяется сформированность у обучающихся компетенций, указанных в разделе 3 настоящей программы. Этапность формирования данных компетенций в процессе освоения образовательной программы определяется последовательным освоением дисциплин (модулей) и прохождением практик, а в процессе освоения дисциплины (модуля) – последовательным достижением результатов освоения содержательно связанных между собой разделов, тем.

Таблица 6 – Соответствие разделов, тем дисциплины (модуля), результатов обучения по дисциплине (модулю) и оценочных средств

Контролируемый раздел, тема дисциплины (модуля)	Код контролируемой компетенции	Наименование оценочного средства
Объектная модель. Классы и объекты	ПК-2	Отчеты о ЛР, устный опрос на экзамене
Основы языка Java	ПК-2	Отчеты о ЛР, устный опрос на экзамене
Механизм классов	ПК-2	Отчеты о ЛР, устный опрос на экзамене
Наследование	ПК-2	Отчеты о ЛР, устный опрос на экзамене
Классы коллекций	ПК-2	Отчеты о ЛР, устный опрос на экзамене
Обработка исключений	ПК-2	Отчеты о ЛР, устный опрос на экзамене
Ввод-вывод и сериализация	ПК-2	Отчеты о ЛР, устный опрос на экзамене

Для оценивания результатов обучения в виде **знаний** используются следующие типы контроля:

- индивидуальное собеседование (устный опрос по ЛПР).
- письменные работы (отчеты о выполнении ЛПР).

Тестовые задания охватывают содержание всего пройденного материала. Индивидуальное собеседование проводится по разработанным вопросам к зачету/экзамену. Письменная работа (отчет о выполнении ЛПР) проводится по отдельному учебному элементу программы дисциплины.

Для оценивания результатов обучения в виде **умений и владений** используются практические контрольные задания (далее – ПКЗ), включающих одну или несколько задач (вопросов) в виде краткой формулировки действий (комплекса действий), которые следует выполнить, или описание результата, который нужно получить.

По сложности ПКЗ разделяются на простые и комплексные задания.

Простые ПКЗ предполагают решение в одно или два действия. К ним можно отнести: простые ситуационные задачи с коротким ответом или простым действием; несложные задания по выполнению конкретных действий. Простые задания применяются для оценки умений. Комплексные задания требуют многоходовых решений как в типичной, так и в нестандартной ситуациях. Это задания в открытой форме, требующие поэтапного решения и развернутого ответа, в т.ч. задания на индивидуальное или коллективное выполнение проектов, на выполнение практических действий или лабораторных работ. Комплексные практические задания применяются для оценки владений.

Типы практических контрольных заданий:

- задания на установление правильной последовательности, взаимосвязанности действий, выяснения влияния различных факторов на результаты выполнения задания;
- установление последовательности (описать алгоритм выполнения действия),
- нахождение ошибок в последовательности (определить правильный вариант последовательности действий);
- указать возможное влияние факторов на последствия реализации умения и т.д.
- задания на принятие решения в нестандартной ситуации (ситуации выбора, многоальтернативности решений, проблемной ситуации);
- задания на оценку последствий принятых решений;
- задания на оценку эффективности выполнения действия.

7.2. Описание показателей и критериев оценивания компетенций, описание шкал оценивания

Таблица 7 – Показатели оценивания результатов обучения в виде знаний

Шкала оценивания	Критерии оценивания
5 «отлично»	демонстрирует глубокое знание теоретического материала, умение обоснованно излагать свои мысли по обсуждаемым вопросам, способность полно, правильно и аргументированно отвечать на вопросы, приводить примеры
4 «хорошо»	демонстрирует знание теоретического материала, его последовательное изложение, способность приводить примеры, допускает единичные ошибки, исправляемые после замечания преподавателя
3 «удовлетворительно»	демонстрирует неполное, фрагментарное знание теоретического материала, требующее наводящих вопросов преподавателя, допускает существенные ошибки в его изложении, затрудняется в приведении примеров и формулировке выводов
2 «неудовлетворительно»	демонстрирует существенные пробелы в знании теоретического материала, не способен его изложить и ответить на наводящие вопросы преподавателя, не может привести примеры

Таблица 8 – Показатели оценивания результатов обучения в виде умений и владений

Шкала оценивания	Критерии оценивания
5 «отлично»	демонстрирует способность применять знание теоретического материала при выполнении заданий, последовательно и правильно выполняет задания, умеет обоснованно излагать свои мысли и делать необходимые выводы
4 «хорошо»	демонстрирует способность применять знание теоретического материала при выполнении заданий, последовательно и правильно выполняет задания, умеет обоснованно излагать свои мысли и делать необходимые выводы, допускает единичные ошибки, исправляемые после замечания преподавателя
3 «удовлетворительно»	демонстрирует отдельные, несистематизированные навыки, испытывает затруднения и допускает ошибки при выполнении заданий, выполняет задание по подсказке преподавателя, затрудняется в формулировке выводов
2 «неудовлетворительно»	не способен правильно выполнить задания

7.3. Контрольные задания и иные материалы, необходимые для оценки результатов обучения по дисциплине (модулю)

Фрагмент лабораторной работы 1 «Основы работы с объектами»

- Откройте в BlueJ проект shapes из каталога \BlueJ\examples. Сохраните проект под именем shapes-<ваша_фамилия>
- Откомпилируйте проект (Tools – Compile)
 - Каково значение поля isVisible?
 - Выполните метод setVisible() для круга. Опишите произошедшие изменения
- Создайте еще один объект класса Circle с именем circle2. Для этого объекта:
 - Откройте окно свойств объекта (Inspect). Почему круг не отображается? Отобразите круг
 - Вызовите метод setColor() и введите значение параметра “red”. Аналогично

измените цвет круга на синий (“blue”). Что произойдет, если вы введете значение цвета без кавычек? Что произойдет, если задать значение цвета, не предусмотренное проектом (например, “cyan”)?

с. Измените размер круга на 10

d. Как сдвинуть круг вверх?

е. Установите круг в левый верхний угол окна. Каковы его координаты?

Результат работы – программа, написанная на языке программирования JAVA.

Фрагмент лабораторной работы 2 «Взаимодействие объектов»

1. Откройте проект shapes из каталога Examples. Сохраните проект под именем shapes2-<ваша_фамилия>.

2. Создайте в классе Circle еще один конструктор, позволяющий задавать при создании круга его координаты. Для этого добавьте в класс Circle после существующего конструктора следующий код:

```
public Circle(int x, int y) {
    diameter = 30;
    xPosition = x;
    yPosition = y;
    color = "blue";
    isVisible = false;
}
```

4. Проверьте работу созданного конструктора, для этого при создании нового круга выберите новый конструктор. Обратите внимание, что BlueJ потребует от вас указать значения параметров x и y.

5. Создайте в классе Circle еще один конструктор, позволяющий указывать при создании круга его координаты, цвет и размер. Протестируйте работу конструктора.

6. Реализуйте аналогичные конструкторы для классов Square (квадрат) и Triangle (треугольник)

7. Методы движения круга в классах фигур moveHorizontal() и moveVertical() сдвигают объект относительно его текущей позиции и не позволяют задавать абсолютное положение объекта. Находясь внутри класса фигуры, можно установить абсолютное местоположение, напрямую установив значения полей xPosition и yPosition. Однако обычные пользователи объекта этой возможности лишены (например, вы не можете сделать это в интерактивном режиме BlueJ. Почему?). Создайте в классе Circle метод moveTo(), который предоставит такую возможность, для этого: а. Вставьте в исходный текст класса Circle следующий код:

```
public void moveTo(int x, int y) {
    ?
}
```

Результат работы – программа, написанная на языке программирования JAVA.

Фрагмент лабораторной работы 3 «Greenfoot. Взаимодействие объектов»

1. Откройте в Greenfoot проект-сценарий crabgame (предварительно скопировав его с сервера из каталога JavaSE\Labs\bluej-projects). Сохраните проект под именем crabgame-<ваша_фамилия>. Нажмите кнопку Compile, чтобы скомпилировать проект.

2. Справа вы видите классы проекта. Классы World и Actor являются стандартными для Greenfoot и представляют, соответственно, базовые классы для создания игрового мира и игровых объектов. В данном проекте игровой мир реализован классом Crabworld, а игровые объекты – классами Animal (животное) и Crab (краб). Эти классы наследуют от базовых классов World и Actor базовые свойства и поведение.

3. Нажмите правой кнопкой на классе Crab и выберите `new Crab()`, чтобы создать объект класса Crab. Поместите созданный объект в игровой мир левой кнопкой мыши.

4. Программа в Greenfoot выполняется следующим образом: для каждого объекта в игровом мире выполняется метод `act()`. Выполнить один шаг программы можно, нажав кнопку Act внизу экрана, запустить программу на выполнение можно кнопкой Run, при этом скорость выполнения можно регулировать с помощью слайдера Speed. Кроме того, вы можете выполнять отдельные методы отдельных объектов, для этого кликните правой кнопкой на объекте и выберите метод, который хотите выполнить. Выполните метод `act()` для созданного вами ранее краба. Что происходит? Запустите программу на выполнение? Какой результат вы наблюдаете?

5. Поскольку класс Crab наследует от класса Animal (а тот, в свою очередь, от класса Actor), он может также выполнять все методы, описанные в родительском классе. Для этого после клика правой кнопкой на объекте выберите пункт `Inherited from Animal` либо `Inherited from Actor`. Выполните метод `move()`. Что происходит? Выполните метод `turn()`. Какой результат? Что определяет параметр этого метода?

6. Откройте исходный текст класса Crab, дважды кликнув на классе (либо кликнув правой кнопкой и выбрав `Open editor`). Вы можете видеть определение класса Crab. Строка `extends Animal` в заголовке класса определяет, что класс Crab наследует от класса Animal. Обратите внимание на содержимое метода `act()` и соотнесите это с поведением краба, наблюдаемым в п. 4.

Результат работы – программа, написанная на языке программирования JAVA

Фрагмент лабораторной работы 4 «Основы наследования»

В данной лабораторной работе вы разработаете игру в жанре `scroll shooter`. Описание игры: управляемый игроком космический корабль движется слева направо, навстречу ему вылетают различные противники, которых нужно уничтожать выстрелами из бортового лазера. Рекомендуется использовать Greenfoot 3.

1. На первом шаге нужно создать и настроить игровой мир

a. Создайте новый проект (сценарий) в Greenfoot, выбрав `New Java Scenario`, и назвав его, например, `SpaceShooter`. Откройте папку сценария в проводнике Windows, и скопируйте в подпапку `images` все файлы изображений из каталога `Java\Java SE\Labs\bluej-projects\sp-shooter-game\images` на сервере

b. Отредактируйте класс `MyWorld`. Нам необходимо, чтобы окно игры было размером 1024 x 768 пикселей, размер ячейки игрового мира – 1 пиксель. Эти параметры задаются в конструкторе игрового мира посредством соответствующего вызова конструктора суперкласса `World`. Кроме того, игровые объекты будут появляться из-за границы экрана и улетать за нее. Игровой мир в Greenfoot по умолчанию является ограниченным, т.е. перемещение игровых объектов за границы экрана невозможно. Чтобы изменить это поведение, добавьте еще один параметр к вызову конструктора суперкласса, указав его значение `false`.

c. Следующий шаг – настройка основы для игровых объектов. Поскольку все объекты в игре будут находиться в движении, желательно весь функционал движения реализовать в отдельном классе, от которого затем будут наследовать все классы объектов.

a. Создайте класс – наследник класса `Actor`, назвав его `MovingObject` (изображение для объекта не задавайте, объекты этого класса непосредственно в игре задействованы не будут).

b. В классе создайте два поля `velocityX` и `velocityY`, задающие текущую скорость движения по оси X и Y соответственно. Значения скорости могут быть положительными либо отрицательными, обеспечивая движение в разных направлениях. Для полей рекомендуется задать спецификатор доступа `protected`, чтобы сделать их доступными для классов-наследников.

Результат работы – программа, написанная на языке программирования JAVA.

Фрагмент лабораторной работы 5 «Базовые конструкции Java»

В данной лабораторной работе вы познакомитесь с работой со встроенными типами данных в Java, с базовыми синтаксическими конструкциями языка, а также начнете использовать среду разработки Eclipse.

В Eclipse, в отличие от BlueJ и Greenfoot, программа выполняется не в интерактивном, а в обычном режиме. Точкой запуска программы является метод `main()` следующего вида:

```
public static void main(String[] args) {
    ... // код запуска программы
}
```

Чтобы программу можно было запустить, подобный метод должен присутствовать хотя бы в одном классе. При этом, поскольку метод является статическим, он имеет определенные ограничения (со статическими методами вы познакомитесь позднее в рамках курса), поэтому не рекомендуется помещать в него большое количество кода. Часто основную программу реализуют в отдельном методе класса, а в методе `main()` создают объект этого класса и запускают выполнение созданного метода. Пример:

```
public class Runner {
    ...
    public void run() {
        ... // основной код программы
    }
    public static void main(String[] args) {
        Runner r = new Runner(); // Создание объекта класса с программой
        r.run(); // Запуск основного метода программы
    }
}
```

В данной работе вам потребуется считывать данные с клавиатуры и выводить их на экран в консольном (текстовом) режиме. Для вывода на экран используется уже знакомая вам конструкция вида:

Результат работы – программа, написанная на языке программирования JAVA.

Фрагмент лабораторной работы 6 «Создание классов»

В данной лабораторной работе вы будете работать над созданием классов различного назначения, будете использовать особые возможности реализации классов, а также познакомитесь с основами реализации графических приложений в Java.

Для создания графических приложений в Java используется встроенная библиотека Swing. Основой для построения графических приложений в Java служит встроенный класс `JFrame`, представляющий собой реализацию графического окна приложения. Окно состоит из области заголовка и рабочей области для размещения компонентов.

Для создания окна необходимо:

Создать объект класса JFrame

```
JFrame frame = new JFrame();
```

Установить размер окна

```
frame.setSize(400, 300);
```

Результат работы – программа, написанная на языке программирования JAVA.

Фрагмент лабораторной работы 7 «Абстрактные классы и интерфейсы»

Импортируйте проект HeaterProject

В проекте класс Runner предназначен для запуска программы, класс HeaterProgram обеспечивает интерфейс управления.

Представленная версия приложения полностью работоспособна, но его внутренняя структура не оптимальна. Требуется применить различные методики абстракции, чтобы улучшить дизайн приложения и обеспечить лучшие возможности его модификации и расширения.

Приложение состоит из следующих классов:

Heater – класс, моделирующий нагреватель

HeaterView – графический интерфейс, имитирующий табло нагревателя; отображает заданную в нагревателе температуру (для отображения используется класс GraphicalDisplay)

GraphicalDisplay - класс, позволяющий выводить до трех текстовых строк с заголовками на графическую форму

Вы будете работать с классами Heater, HeaterView и HeaterProgram. Класс GraphicalDisplay будет использоваться как готовый и модифицироваться не будет.

Результат работы – программа, написанная на языке программирования JAVA.

Примерные темы для курсовых работ по дисциплине

1. Java Редактор для рисования электрических схем.
2. Разработка распределенного Web-приложения на Java.
3. Разработка сайта школьной библиотеки на технологии HTML и JavaScript.
4. Интеллектуальный бот для игры "Крестики-нолики" на Java.
5. Использование JAVA -технологий для разработки графических приложений
6. Программирование на Java в телекоммуникационных сетях.

Перечень вопросов и заданий, выносимых на экзамен

1. Метод getTotalBoxes вычисляет и возвращает общее количество всех упаковок всех товаров во всех заказах в общем заказе. Если общий заказ не содержит заказов, метод возвращает 0. Реализуйте метод

2. метод getProductWithMaxBoxes возвращает наименование товара с максимальным количеством упаковок в заказе. При наличии нескольких товаров с одинаковым максимальным количеством упаковок метод может вернуть наименование любого из них. Реализуйте метод getProductWithMaxBoxes.

3. Реализуйте класс `Line`. Ваша реализация должна включать конструктор с тремя целочисленными параметрами, представляющими a , b и c . Можете считать, что значения параметров a и b не равны 0. Класс должен включать метод `getSlope()`, вычисляющий и возвращающий уклон прямой, и метод `isOnLine` с двумя целочисленными параметрами x и y , возвращающий `true`, если точка с координатами x и y находится на прямой, и `false` в противном случае. Ваш класс должен использоваться и выдавать результаты в соответствии с приведенным в описании примером кода. Переполнение целых чисел можно игнорировать.

4. Реализуйте метод (в некотором произвольном стороннем классе) `getCubeTosses`, который принимает параметрами экземпляр кубика и количество бросков. Метод должен возвращать массив значений, полученных посредством выполнения заданного количества бросков заданного кубика. Параметры метода: `cube` – экземпляр класса `NumberCube`, выполняющего броски; `numTosses` – количество бросков, которые необходимо выполнить (`numTosses > 0`).

5. Реализуйте метод `getShortestLayover`, вычисляющий наименьшую длительность пересадки. Время пересадки – количество минут между прибытием рейса и временем отправления следующего за ним. Если в поездке имеется два и более рейса, метод возвращает наименьшую длительность пересадки, в противном случае возвращает `-1`.

6. Класс `SubstringChecker` принимает любую строку, содержащую определенную подстроку. Например, нижеследующий объект `broccoliChecker` класса `SubstringChecker` принимает все строки, содержащие подстроку “брокколи”.

7. Метод `removeProductName` обновляет общий заказ, удаляя из него все заказы, в которых наименование товара совпадает с параметром `productVar`. Общий заказ может содержать 0 или более заказов с наименованием, совпадающим с `productVar`. Метод должен возвращать общее количество упаковок товаров, удаленных из общего заказа.

8. Реализуйте метод `getCriticalFlights`, список «Критичных по времени» рейсов. Критичным рейсом считается такой, время пересадки после которого менее заданного параметром `criticalTime`. Время пересадки – количество минут между прибытием рейса и временем отправления следующего за ним. Если в поездке имеется два и более рейса, метод возвращает список критичных по времени рейсов, в противном случае возвращает `null`.

9. Реализуйте метод `longestLevelTrailSegment` класса `Trail`. Метод должен возвращать максимальную длину ровного сегмента в заданном маршруте, либо `-1`, если в маршруте нет 17 ровных сегментов (минимальная длина сегмента равна двум; считайте, что маршрут содержит как минимум две отметки). Сегмент маршрута определяется начальной отметкой, конечной отметкой и всеми отметками между ними. Сегмент маршрута является ровным, если разница между максимальной высотой и минимальной высотой в сегменте меньше либо равна 10. Для маршрута, приведенного в описании задачи, существует несколько ровных сегментов, например, между отметками 4 и 5 (длина 2), между отметками 7 и 8 (длина 2), 10 и 11 (длина 2) и т.д.. При этом наиболее длинным ровным сегментом будет являться сегмент между отметками 7 и 10, его длина равна 4. Таким образом, для приведенного набора данных метод `longestLevelTrailSegment` должен вернуть значение 4.

10. Реализуйте метод `getLayoverDuration`, вычисляющий общую длительность нахождения в аэропортах во время поездки. Нахождением в аэропорту считать время между прибытием рейса и отправлением следующего рейса. Если в поездке имеется два и более рейса, метод возвращает общую длительность нахождения в аэропортах,

Таблица 9 – Примеры оценочных средств с ключами правильных ответов

№ п/п	Тип задания	Формулировка задания	Правильный ответ	Время выполнения (в минутах)
ПК-2	Способен разрабатывать программное обеспечение, включая проектирование, отладку, проверку работоспособности и модификацию ПО.			

№ п/п	Тип задания	Формулировка задания	Правильный ответ	Время выполнения (в минутах)
1.	Задание закрытого типа	Какой класс является родительским для всех классов в Java? 1) Object 2) String 3) Integer 4) ArrayList	1	2
2.		Какой метод используется для вывода текста на консоль? 1) System.out.println() 2) System.out.printnln() 3) System.out.write() 4) System.out.read()	2	2
3.		Какой ключевое слово используется для создания нового объекта? 1) new 2) create 3) instance 4) object	1	2
4.		Какой цикл используется для выполнения блока кода хотя бы один раз? 1) for 2) while 3) do-while 4) foreach	3	2
5.		Какой будет вывод программы? int[] numbers = {1, 2, 3, 4, 5}; int sum = 0; for (int i = 0; i < numbers.length; i++) { if (numbers[i] % 2 == 0) { sum += numbers[i]; } } System.out.println(sum); 1) 6 2) 9 3) 15 4) 20	1	2
6.	Задание открытого типа	Объясните, что такое инкапсуляция в контексте объектно-ориентированного программирования (ООП) и как она реализуется в Java. Объясните, что такое инкапсуляция в контексте объектно-ориентированного	Инкапсуляция — это один из ключевых принципов ООП, который заключается в скрытии внутреннего состояния объекта и предоставлении доступа к нему только	5

№ п/п	Тип задания	Формулировка задания	Правильный ответ	Время выполнения (в минутах)
		программирования (ООП) и как она реализуется в Java.	через публичные методы. В Java инкапсуляция реализуется с помощью модификаторов доступа, таких как private, protected и public. Например, поля класса можно объявить как private, а методы доступа (геттеры и сеттеры) — как public, чтобы контролировать доступ к данным и обеспечить их целостность.	
7.		Опишите, как работает механизм наследования в Java, и приведите пример.	Наследование в Java позволяет одному классу (подклассу) наследовать поля и методы другого класса (суперкласса). Это способствует повторному использованию кода и созданию иерархий классов. Наследование реализуется с помощью ключевого слова extends. Например, если у нас есть класс Animal с методом makeSound(), то класс Dog может наследовать от Animal и переопределять метод makeSound(), чтобы он выводил "Woof!".	5
8.		Что такое полиморфизм и как он реализуется в Java? Приведите пример.	Полиморфизм — это способность объектов разных классов быть обработанными через общий интерфейс. В Java полиморфизм реализуется через наследование и интерфейсы.	5

№ п/п	Тип задания	Формулировка задания	Правильный ответ	Время выполнения (в минутах)
			Например, если у нас есть интерфейс Shape с методом draw(), то классы Circle и Square могут реализовать этот интерфейс и предоставлять свои собственные версии метода draw(). Это позволяет использовать объекты Circle и Square через ссылку типа Shape, вызывая соответствующий метод draw().	
9.		Объясните, что такое исключения в Java и как они обрабатываются.	Исключения в Java — это механизм обработки ошибок и необычных ситуаций, возникающих во время выполнения программы. Исключения позволяют разделить нормальный поток выполнения программы и код, обрабатывающий ошибки. В Java исключения обрабатываются с помощью блоков try, catch и finally. В блоке try размещается код, который может вызвать исключение, в блоке catch — код, обрабатывающий исключение, а в блоке finally — код, который выполняется в любом случае, независимо от того, было ли исключение.	5
10.		Опишите, как работает сборка мусора (Garbage Collection) в Java.	Сборка мусора (Garbage Collection) в Java — это автоматический	5

№ п/п	Тип задания	Формулировка задания	Правильный ответ	Время выполнения (в минутах)
			процесс управления памятью, который освобождает память, занятую объектами, которые больше не используются программой. Этот процесс управляется виртуальной машиной Java (JVM) и позволяет разработчикам не беспокоиться о ручном управлении памятью. JVM использует различные алгоритмы для определения, какие объекты больше не доступны, и освобождения памяти, занятой этими объектами. Это помогает предотвратить утечки памяти и обеспечивает эффективное использование ресурсов.	

Полный комплект оценочных материалов по дисциплине (модулю) (фонд оценочных средств) хранится в электронном виде на кафедре, утверждающей рабочую программу дисциплины (модуля), и в Центре мониторинга и аудита качества обучения.

7.4. Методические материалы, определяющие процедуры оценивания результатов обучения по дисциплине (модулю)

Дисциплина «Технологии Java» изучается студентами 3 курса в течение 5 семестра. Форма аттестации по дисциплине в 5 семестре – «экзамен».

Итоговая оценка по промежуточной аттестации выставляется в соответствии с Положением АГУ о балльно-рейтинговой системе (БАРС). Итоговая оценка складывается из баллов, полученных студентами за текущую успеваемость в течении семестра и баллов, полученных студентом на зачетном экзамене. Для получения положительной оценки студенту необходимо набрать минимально 60 баллов.

Экзамен проходит в форме практического задания, примеры составленных заданий (п. 7.3). Один билет включает в себя одно задание. Выбор билета осуществляется в случайном порядке. На подготовку студенту отводится не менее 40 мин. Во время проведения экзамена студенту запрещено пользоваться сотовым телефоном и иными средствами связи, персональным компьютером, сетью Интернет, заготовленными заранее ответами и т.п. Студент, получивший замечание в использовании вышеперечисленного удаляется с экзамена с выставлением 0 баллов. Во время защиты к устному ответу студенты могут делать записи на чистом листе, а затем взять их для ответа. Во время устного ответа

чтение текста, написанного при подготовке к устному ответу НЕ ДОПУСТИМО. Такой ответ будет оценен в 0 баллов.

Преподаватель, реализующий дисциплину (модуль), в зависимости от уровня подготовленности обучающихся может использовать иные формы, методы контроля и оценочные средства, исходя из конкретной ситуации.

Таблица 10 – Технологическая карта рейтинговых баллов по дисциплине (модулю)

№ п/п	Контролируемые мероприятия	Количество мероприятий / баллы	Максимальное количество баллов	Срок представления
Основной блок				
1.	<i>Ответ на занятия</i>	6/2	12	В соотв. с табл. 2
2.	<i>Выполнение лабораторной работы</i>	7/4	28	
Всего			40	-
Блок бонусов				
3.	<i>Посещение занятий без пропуска</i>		3	
4.	<i>Своевременное выполнение всех заданий</i>		3	
5.	<i>Активность студента на занятии</i>		4	
Всего			10	-
Дополнительный блок**				
6.	<i>Экзамен</i>		50	
Всего			50	-
ИТОГО			100	-

[Примечание: * – для дисциплины (модуля) с итоговой формой контроля «Зачёт» / «Дифференцированный зачёт», ** – для дисциплины (модуля) с итоговой формой контроля «Экзамен»]

Таблица 11 – Система штрафов (для одного занятия)

Показатель	Балл
<i>Опоздание на занятие</i>	-1
<i>Нарушение учебной дисциплины</i>	-1
<i>Неготовность к занятию</i>	-2
<i>Пропуск занятия без уважительной причины</i>	-2

Таблица 12 – Шкала перевода рейтинговых баллов в итоговую оценку за семестр по дисциплине (модулю)

Сумма баллов	Оценка по 4-балльной шкале	
90–100	5 (отлично)	Зачтено
85–89	4 (хорошо)	
75–84		
70–74		
65–69	3 (удовлетворительно)	
60–64		
Ниже 60	2 (неудовлетворительно)	Не зачтено

При реализации дисциплины (модуля) в зависимости от уровня подготовленности обучающихся могут быть использованы иные формы, методы контроля и оценочные средства, исходя из конкретной ситуации.

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

8.1. Основная литература

1. Царёв Р.Ю., Алгоритмы и структуры данных (CDIO) : учебник / Царёв Р.Ю. - Красноярск : СФУ, 2016. - 204 с. - ISBN 978-5-7638-3388-1 - Текст : электронный // ЭБС "Консультант студента" : [сайт]. – URL <http://www.studentlibrary.ru/book/ISBN9785763833881.html>.
2. Златопольский Д.М., Программирование: типовые задачи, алгоритмы, методы / Златопольский Д.М. - М. : БИНОМ, 2012. - 223 с. - ISBN 978-5-9963-0888-0 - Текст : электронный // www.studentlibrary.ru/book/ISBN9785001017899.html.

8.2. Дополнительная литература

1. Свистунов А.Н., Построение распределенных систем на Java / Свистунов А.Н. - М.: Национальный Открытый Университет "ИНТУИТ", 2016. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : http://www.studentlibrary.ru/book/intuit_251.html.
2. Жолобов Д.А., Жарких Л.И, Морозов Б.Б. Программирование на языках высокого уровня: учебное пособие /. — Издательский дом «Астраханский университет», 2015. — 128 с.
3. Монахов В.В., Язык программирования Java и среда NetBeans / Монахов В.В. - М. : Национальный Открытый Университет "ИНТУИТ", 2016. - Текст : электронный // ЭБС "Консультант студента" : [сайт]. - URL : http://www.studentlibrary.ru/book/intuit_419.html.

8.3. Интернет-ресурсы, необходимые для освоения дисциплины (модуля)

1. Электронная библиотечная система (ЭБС) ООО «Политехресурс» «Консультант студента»: www.studentlibrary.ru.

9. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ (МОДУЛЯ)

Для проведения лекционных занятий необходима мультимедийная аудитория, оснащенная рабочими местами студентов. Для проведения лабораторных занятий необходима аудитория, оснащенная компьютерными рабочими местами студентов и доступом в Интернет.

Рабочая программа дисциплины (модуля) при необходимости может быть адаптирована для обучения (в том числе с применением дистанционных образовательных технологий) лиц с ограниченными возможностями здоровья, инвалидов. Для этого требуется заявление обучающихся, являющихся лицами с ограниченными возможностями здоровья, инвалидами, или их законных представителей и рекомендации психолого-медико-педагогической комиссии. Для инвалидов содержание рабочей программы дисциплины (модуля) может определяться также в соответствии с индивидуальной программой реабилитации инвалида (при наличии).