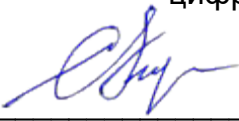


МИНОБРНАУКИ РОССИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «ВГУ»)

**УТВЕРЖДАЮ**

Заведующий кафедрой  
цифровых технологий

 / Кургалин С.Д.

22.04.2024 г.

**РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ**

Б1.В.ДВ.01.01.03 ЯЗЫК ПРОГРАММИРОВАНИЯ PYTHON

**1. Код и наименование направления подготовки/специальности:**

02.03.01 Математика и компьютерные науки

**2. Профиль подготовки/специализация:**

математическое и программное обеспечение информационных систем и технологий

**3. Квалификация выпускника: бакалавр**

**4. Форма обучения: очная**

**5. Кафедра, отвечающая за реализацию дисциплины: цифровых технологий**

**6. Составители программы:**

Романов Александр Викторович, ст. преподаватель

**7. Рекомендована: НМС ФКН (протокол № 5 от 05.03.2024)**

**8. Учебный год: 2026-2027 Семестры: 6**

## 9. Цели и задачи учебной дисциплины

- дать представление об основных возможностях языка программирования Python,
- дать обзор основных типов и средств программирования на языке Python,
- рассмотреть вопрос документирования и сопровождения программного кода, написанного на языке Python.
- изучить вопросы модульного построения программ,
- изучить модели функционирования языка программирования Python.

## 10. Место учебной дисциплины в структуре ООП:

дисциплина относится к вариативной части блока Б1. Для успешного освоения дисциплины необходимо предварительное изучение математического анализа и программирования.

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

Код	Название компетенции	Код(ы)	Индикатор(ы)	Планируемые результаты обучения
ПК-1	Способен демонстрировать базовые знания математических и естественных наук, основ программирования и информационных технологий.	ПК-1.1	Обладает базовыми знаниями, полученными в области математических и (или) естественных наук, программирования и информационных технологий	Знать: методы и средства языка программирования Python, основы объектной модели языка, документирования и сопровождения программного кода;
ПК-1	Способен демонстрировать базовые знания математических и естественных наук, основ программирования и информационных технологий.	ПК-1.2	Умеет находить, формулировать и решать стандартные задачи в собственной научно-исследовательской деятельности в математике и информатике	Уметь: использовать средства программирования языка Python с учетом модульного построения программ и объектной модели языка;
ПК-1	Способен демонстрировать базовые знания математических и естественных наук, основ программирования и информационных технологий.	ПК-1.3	Имеет практический опыт научно-исследовательской деятельности в математике и информатике	Владеть: практическими навыками решения вычислительных задач с помощью языка программирования Python.

12. Объем дисциплины в зачетных единицах/час. — 3/108.

Форма промежуточной аттестации 6 семестр – зачёт с оценкой.

### 13. Трудоемкость по видам учебной работы

Вид учебной работы		Трудоемкость	
		Всего	По семестрам
			6 сем.
Аудиторные занятия		66	66
в том числе:	лекции	34	34
	практические	16	16
	лабораторные	16	16
Самостоятельная работа		42	42
Зачёт с оценкой			
Итого:		108	108

#### 13.1. Содержание дисциплины

№ п/п	Наименование раздела дисциплины	Содержание раздела дисциплины	Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК*
<b>1. Лекции</b>			
1.1	Введение в язык программирования Python	Введение. Области применения языка программирования Python: системное программирование, графические пользовательские интерфейсы, сценарии для Интернета, интеграция компонентов, программирование для баз данных, быстрое прототипирование, численное и научное программирование. Особенности синтаксиса, переносимость и базовые концепции.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.2	Введение в интерпретатор Python	Выполнение программ. Компиляция в байт-код и виртуальная машина Python (PVM). Альтернативные реализации Python. Инструменты оптимизации выполнения. Фиксированные двоичные файлы. Интерактивная подсказка. Командная строка системы и файлы. Исполняемые сценарии в стиле Unix: #! Импортирование и повторная загрузка модулей.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.3	Типы объектов Python	Концептуальная иерархия Python. Основные типы данных Python: числа, строки, списки, словари, кортежи, файлы	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.4	Числовые типы	Основы числовых типов. Числовые литералы. Операции выражений Python. Деление: классическое, с округлением в меньшую сторону и настоящее. Комплексные числа. Шестнадцатеричная, восьмеричная и двоичная формы записи: литералы и преобразования. Десятичные типы. Дробный тип. Множества и включения множеств. Булевские значения.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.5	Динамическая типизация	Динамическая типизация и разделяемые ссылки. Разделяемые ссылки и изменения на месте.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.6	Основы строк	Строковые литералы и операции. Строки в одинарных и двойных кавычках. Управляющие последовательности. Неформатированные строки. Многострочные блочные строки. Индексация и нарезание. Инструменты преобразования строк. Строковые методы. Выражения форматирования строк. Методы форматирования строк.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>

1.7	Списки и словари	Списки. Списковые литералы и операции. Базовые списковые операции. Итерация по спискам и списковые включения. Индексация, нарезание и матрицы. Изменение списков на месте. Распространенные списковые методы. Словари Словарные литералы и операции. Изменение словарей на месте. Дополнительные словарные методы. Вложение словарей.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.8	Кортежи и файлы	Кортежи. Литералы и операции над кортежами. Именованные кортежи. Файлы. Файловые операции.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.9	Введение в операторы Python	Операторы Python. Операторы присваивания. Расширенная распаковка последовательностей в Python. Групповые присваивания. Операторы if. Множественное ветвление. Значения истинности и булевские проверки. Тернарное выражение if/else. Циклы while. Операторы break, continue, pass и конструкция else цикла. Циклы for. Циклы с подсчетом: range. Параллельные обходы: zip и map. Итерации и включения. Документация.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.10	Функции	Операторы и выражения, связанные с функциями. Операторы def. Полиморфизм в Python. Основы областей видимости в Python. Оператор global. Области видимости и вложенные функции. Фабричные функции: замыкания. Основы передачи аргументов. Аргументы и разделяемые ссылки. Формы сопоставления аргументов функций. Списковые включения.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.11	Модули и пакеты	Создание модулей. Импортирование и атрибуты. Оператор import. Оператор from. Файлы байт-кода: русаче. Пространства имен модулей. Перезагрузка модулей. Пакеты модуле. Пакеты пространств имен.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.12	Классы и объектно-ориентированное программирование	Поиск в иерархии наследования. Классы и экземпляры. Вызовы методов. Создание деревьев классов. Перегрузка операций. Методики связывания классов. Абстрактные суперклассы. Вложенные классы.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
1.13	Исключения	Роли, исполняемые исключениями. Стандартный обработчик исключений. Перехват исключений. Генерация исключений. Исключения, определяемые пользователем. Оператор try/except/finally. Оператор raise. Оператор assert. Диспетчеры контекстов with/as.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
<b>2. Практические занятия</b>			
2.1	Введение в язык программирования Python	Введение. Области применения языка программирования Python: системное программирование, графические пользовательские интерфейсы, сценарии для Интернета, интеграция компонентов, программирование для баз данных, быстрое прототипирование, численное и научное программирование. Особенности синтаксиса, переносимость и базовые концепции.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.2	Введение в интерпретатор Python	Выполнение программ. Компиляция в байт-код и виртуальная машина Python (PVM). Альтернативные реализации Python. Инструменты оптимизации выполнения. Фиксированные двоичные файлы. Интерактивная подсказка. Командная строка системы и файлы. Исполняемые сценарии в стиле Unix: #! Импортирование и повторная загрузка модулей.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.3	Типы объектов Python	Концептуальная иерархия Python. Основные типы данных Python: числа, строки, списки, словари, кортежи, файлы	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>

2.4	Числовые типы	Основы числовых типов. Числовые литералы. Десятичные типы. Дробный тип. Множества и включения множеств. Булевские значения.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.5	Динамическая типизация	Динамическая типизация и разделяемые ссылки. Разделяемые ссылки и изменения на месте.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.6	Основы строк	Строковые литералы и операции. Строки в одинарных и двойных кавычках. Управляющие последовательности. Неформатированные строки. Многострочные блочные строки. Индексация и нарезание. Инструменты преобразования строк. Строковые методы. Выражения форматирования строк. Методы форматирования строк.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.7	Списки и словари	Списковые литералы и операции. Базовые списковые операции. Итерация по спискам и списковые включения. Индексация, нарезание и матрицы. Изменение списков на месте. Распространенные списковые методы. Словарные литералы и операции. Изменение словарей на месте. Дополнительные словарные методы. Вложение словарей.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.8	Кортежи и файлы	Литералы и операции над кортежами. Именованные кортежи. Файловые операции.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.9	Введение в операторы Python	Операторы присваивания. Расширенная распаковка последовательностей в Python. Групповые присваивания. Операторы if. Множественное ветвление. Значения истинности и булевские проверки. Тернарное выражение if/else. Циклы while. Операторы break, continue, pass и конструкция else цикла. Циклы for. Циклы с подсчетом: range. Параллельные обходы: zip и map. Итерации и включения. Документация.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.10	Функции	Операторы def. Оператор global. Области видимости и вложенные функции. Фабричные функции: замыкания. Основы передачи аргументов. Аргументы и разделяемые ссылки. Формы сопоставления аргументов функций. Списковые включения.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.11	Модули и пакеты	Создание модулей. Импорт и атрибуты. Оператор import. Оператор from. Файлы байт-кода: русаче. Пространства имен модулей. Перегрузка модулей. Пакеты модуле. Пакеты пространств имен.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.12	Классы и объектно-ориентированное программирование	Поиск в иерархии наследования. Вызовы методов. Создание деревьев классов. Перегрузка операций. Методики связывания классов. Абстрактные суперклассы. Вложенные классы.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>
2.13	Исключения	Перехват исключений. Генерация исключений. Исключения, определяемые пользователем. Оператор try/except/finally. Оператор raise. Оператор assert. Диспетчеры контекстов with/as.	<a href="https://edu.vsu.ru/course/view.php?id=5443">https://edu.vsu.ru/course/view.php?id=5443</a>

### 13.2. Темы (разделы) дисциплины и виды занятий

№ п/п	Наименование темы (раздела) дисциплины	Виды занятий (количество часов)				Всего
		Лекции	Практические	Лабораторные	Самостоятельная работа	
1	Введение в язык программирования Python	2	1	1	2	6
2	Введение в интерпретатор Python	2	1	1	2	6

3	Типы объектов Python	3	1	1	3	8
4	Числовые типы	3	2	2	3	10
5	Динамическая типизация	2	1	1	2	6
6	Основы строк	3	1	1	3	8
7	Списки и словари	3	2	2	3	10
8	Кортежи и файлы	3	2	2	3	10
9	Введение в операторы Python	2	1	1	5	9
10	Функции	3	1	1	3	8
11	Модули и пакеты	3	1	1	3	8
12	Классы и объектно-ориентированное программирование	2	1	1	5	9
13	Исключения	3	1	1	5	10
	Итого:	34	16	16	42	108

#### 14. Методические указания для обучающихся по освоению дисциплины:

Освоение дисциплины складывается из аудиторной работы (учебной деятельности, выполняемой под руководством преподавателя) и внеаудиторной работы (учебной деятельности, реализуемой обучающимся самостоятельно).

Аудиторная работа состоит из работы на лекциях и выполнения практических (или лабораторных) заданий в объеме, предусмотренном учебным планом. Лекция представляет собой последовательное и систематическое изложение учебного материала, направленное на знакомство обучающихся с основными понятиями и теоретическими положениями изучаемой дисциплины. Лекционные занятия формируют базу для практических (или лабораторных) занятий, на которых полученные теоретические знания применяются для решения конкретных практических задач. Обучающимся для успешного освоения дисциплины рекомендуется вести конспект лекций и практических (лабораторных) занятий.

Самостоятельная работа предполагает углублённое изучение отдельных разделов дисциплины с использованием литературы, рекомендованной преподавателем, а также конспектов лекций, презентационным материалом (при наличии) и конспектов практических (лабораторных) занятий. В качестве плана для самостоятельной работы может быть использован раздел 13.1 настоящей рабочей программы, в котором зафиксированы разделы дисциплины и их содержание. В разделе 13.2 рабочей программы определяется количество часов, отводимое на самостоятельную работу по каждому разделу дисциплины. Больше количество часов на самостоятельную работу отводится на наиболее трудные разделы дисциплины. Для самостоятельного изучения отдельных разделов дисциплины используется перечень литературы и других ресурсов, перечисленных в пунктах 15 и 16 настоящей рабочей программы.

Успешность освоения дисциплины определяется систематичностью и глубиной аудиторной и внеаудиторной работы обучающегося.

При использовании дистанционных образовательных технологий и электронного обучения выполнять все указания преподавателей, вовремя подключаться к online занятиям, ответственно подходить к заданиям для самостоятельной работы.

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

а) основная литература:

№ п/п	Источник
1	Саммерфилд, М. Python на практике : учебное пособие / Саммерфилд М. – Москва : ДМК-пресс, 2014. – 338 с. – <a href="https://www.studentlibrary.ru/book/ISBN9785970600955.html">https://www.studentlibrary.ru/book/ISBN9785970600955.html</a>
2	Шелудько, В.М. Основы программирования на языке высокого уровня Python : учебное пособие

	/ Шелудько В.М. – Москва : ЮФУ, 2017. – 146 с. – <a href="https://www.studentlibrary.ru/book/ISBN9785927526499.html">https://www.studentlibrary.ru/book/ISBN9785927526499.html</a>
3	Шелудько, В.М. Язык программирования высокого уровня Python. Функции, структуры данных, дополнительные модули : учебное пособие / Шелудько В.М. – Москва : ЮФУ, 2017. – 107 с. – <a href="https://www.studentlibrary.ru/book/ISBN9785927526482.html">https://www.studentlibrary.ru/book/ISBN9785927526482.html</a>

б) дополнительная литература:

№ п/п	Источник
1	Златопольский, Д.М. Основы программирования на языке Python : учебник / Златопольский Д.М. – Москва : ДМК-пресс, 2017. – 284 с. – <a href="https://www.studentlibrary.ru/book/ISBN9785970605523.html">https://www.studentlibrary.ru/book/ISBN9785970605523.html</a>
2	Хилл, К. Научное программирование на Python [Электронный ресурс] / Хилл К. – Москва : ДМК Пресс, 2021/ – 646 с. – Книга из коллекции ДМК Пресс - Информатика <a href="https://e.lanbook.com/book/241031">https://e.lanbook.com/book/241031</a>
3	Северанс, Ч. Р. Python для всех [Электронный ресурс] / Северанс Ч. Р. – Москва : ДМК Пресс, 2022. – 262 с. – Книга из коллекции ДМК Пресс - Информатика <a href="https://e.lanbook.com/book/241115">https://e.lanbook.com/book/241115</a>
4	Уилкс, М. Профессиональная разработка на Python [Электронный ресурс] / Уилкс М. – Москва : ДМК Пресс, 2021. – 502 с. – Книга из коллекции ДМК Пресс - Информатика <a href="https://e.lanbook.com/book/241121">https://e.lanbook.com/book/241121</a>
5	Русина, Л. Г. Вычислительная математика. Численные методы интегрирования и решения дифференциальных уравнений и систем [Электронный ресурс] / Русина Л. Г. – 2-е изд., стер. Санкт-Петербург : Лань, 2022. – 168 с. – Книга из коллекции Лань - Математика <a href="https://e.lanbook.com/book/195521">https://e.lanbook.com/book/195521</a>
6	Златопольский, Д. М. Основы программирования на языке Python [Электронный ресурс] / Златопольский Д. М. 2-ое изд., испр. и доп. – Москва : ДМК Пресс, 2018. – 396 с. – Книга из коллекции ДМК Пресс - Информатика <a href="https://e.lanbook.com/book/131683">https://e.lanbook.com/book/131683</a>
7	Груздев, А. В. Изучаем Pandas [Электронный ресурс] / Груздев А. В., Хейдт М. 2-ое изд., испр. и доп. – Москва : ДМК Пресс, 2019. – 700 с. – Книга из коллекции ДМК Пресс - Информатика <a href="https://e.lanbook.com/book/131693">https://e.lanbook.com/book/131693</a>
8	Дауни, А. Б. Изучение сложных систем с помощью Python [Электронный ресурс] / Дауни А. Б. – Москва : ДМК Пресс, 2019. – 160 с. – Книга из коллекции ДМК Пресс - Информатика <a href="https://e.lanbook.com/book/131701">https://e.lanbook.com/book/131701</a>
9	Саммерфилд, М. Python на практике [Электронный ресурс] / Саммерфилд М. ; Пер. с англ. Слинкин А.А. – Москва : ДМК Пресс, 2014. – 338 с. – Книга из коллекции ДМК Пресс - Информатика <a href="http://e.lanbook.com/books/element.php?pl1_id=66480">http://e.lanbook.com/books/element.php?pl1_id=66480</a>
10	Лучано, Р. Python. К вершинам мастерства [Электронный ресурс] / Лучано Р. ; Пер. с англ. Слинкин А.А. – Москва : ДМК Пресс, 2016. – 768 с. – Книга из коллекции ДМК Пресс - Информатика <a href="https://e.lanbook.com/book/93273">https://e.lanbook.com/book/93273</a>

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

№ п/п	Ресурс
1	Официальный сайт разработчика. – URL: <a href="https://www.python.org">https://www.python.org</a> (дата обращения: 01.06.2021).
2	Разработка программного обеспечения, технологии и наука. – URL: <a href="https://devpractice.ru/python-lessons">https://devpractice.ru/python-lessons</a> (дата обращения: 01.06.2021).
3	Учебный курс «Программирование на языке высокого уровня (Python)». – URL: <a href="https://www.yuripetrov.ru/edu/python/index.html">https://www.yuripetrov.ru/edu/python/index.html</a> (дата обращения: 01.06.2021).
4	ЗНБ ВГУ: <a href="https://lib.vsu.ru/">https://lib.vsu.ru/</a>
5	Электронно-библиотечная система "Университетская библиотека online": <a href="http://biblioclub.ru/">http://biblioclub.ru/</a>
6	Электронно-библиотечная система "Лань": <a href="https://e.lanbook.com/">https://e.lanbook.com/</a>

**16. Перечень учебно-методического обеспечения для самостоятельной работы**

№ п/п	Источник
1	Бизли, Д. Python. Книга рецептов [Электронный ресурс] / Бизли Д., Джонс Б. К. – Москва : ДМК Пресс, 2019. – 646 с. – Книга из коллекции ДМК Пресс - Информатика <a href="https://e.lanbook.com/book/131723">https://e.lanbook.com/book/131723</a>
2	Стивенсон, Б. Python. Сборник упражнений [Электронный ресурс] / Стивенсон Б. – Москва : ДМК Пресс, 2021. – 238 с. – Книга из коллекции ДМК Пресс - Информатика <a href="https://e.lanbook.com/book/241025">https://e.lanbook.com/book/241025</a>

## 17. Образовательные технологии, используемые при реализации учебной дисциплины, включая дистанционные образовательные технологии (ДОТ, электронное обучение (ЭО), смешанное обучение):

При реализации дисциплины могут использоваться технологии электронного обучения и дистанционные образовательные технологии на базе портала edu.vsu.ru, а также другие доступные ресурсы сети Интернет.

## 18. Материально-техническое обеспечение дисциплины:

Аудитория для лекционных занятий: мультимедиа-проектор, экран для проектора, компьютер с выходом в сеть «Интернет». Специализированная мебель (столы ученические, стулья, доска). Программное обеспечение: LibreOffice v.5-7, программа для просмотра файлов формата pdf, браузер.

Компьютерный класс: специализированная мебель, персональные компьютеры на базе i3-9100-3,6ГГц, мониторы ЖК 19» (30 шт.), мультимедийный проектор, экран.

ПО: ОС Windows v.7, 8, 10, Набор утилит интерпретатор языка CPython, интерпретатор языка Anaconda, IDE PyCharm, редактор Jupiter.

## 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)	Компетенции	Индикаторы достижения компетенции	Оценочные средства
1.	Разделы 1-12	ПК-1	ПК-1.1 ПК-1.2 ПК-1.3	Практические задания, лабораторные работы
Промежуточная аттестация форма контроля – зачёт с оценкой				Практическое задание

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

### 20.1. Текущий контроль успеваемости

Контроль успеваемости по дисциплине осуществляется с помощью тестовых заданий и лабораторных работ.

### Перечень заданий к практическим занятиям

#### Практическое задание №1

1. Распаковка последовательности в отдельные переменные.
2. Распаковка элементов из последовательностей произвольной длины.
3. Оставляем N последних элементов.
4. Поиск N максимальных и минимальных элементов.
5. Реализация очереди с приоритетом.
6. Отображение ключей на несколько значений в словаре.
7. Поддержание порядка в словарях.
8. Вычисления со словарями.
9. Поиск общих элементов в двух словарях.



10. Присваивание имен срезам.
11. Определение наиболее часто встречающихся элементов в последовательности.
12. Сортировка списка словарей по общему ключу.
13. Сортировка объектов, не поддерживающих сравнение.
14. Группирование записей на основе полей.
15. Фильтрация элементов последовательности.
16. Извлечение подмножества из словаря.
17. Отображение имен на последовательность элементов.
18. Одновременное преобразование и сокращение (свертка) данных.
19. Объединение нескольких отображений в одно.

### **Практическое задание №2**

1. Разрезание строк различными разделителями.
2. Поиск текста в начале и в конце строки.
3. Поиск строк с использованием масок оболочки (shell).
4. Поиск совпадений и поиск текстовых паттернов.
5. Поиск и замена текста.
6. Поиск и замена текста без учета регистра.
7. Определение регулярных выражений для поиска кратчайшего совпадения.
8. Написание регулярного выражения для многострочных шаблонов.
9. Нормализация текста в Unicode к стандартному представлению.
10. Использование символов Unicode в регулярных выражениях.
11. Срезание нежелательных символов из строк.
12. Чистка строк.
13. Выравнивание текстовых строк.
14. Объединение и конкатенация строк.
15. Интерполяция переменных в строках.
16. Разбивка текста на фиксированное количество колонок.

### **Практическое задание №3**

1. Округление числовых значений.
2. Выполнение точных вычислений с десятичными дробями.
3. Форматирование чисел для вывода.
4. Работа с бинарными, восьмеричными и шестнадцатеричными целыми числами.
5. Упаковка и распаковка больших целых чисел из байтовых строк.
6. Вычисления с комплексными числами.
7. Работа с бесконечными значениями и NaN.
8. Вычисления с дробями.
9. Вычисления на больших массивах чисел.
10. Вычисления с матрицами и линейная алгебра.
11. Случайный выбор.
12. Перевод дней в секунды и другие базовые методы конвертации времени.
13. Определение даты последней пятницы.
14. Поиск диапазона дат для текущего месяца.
15. Конвертирование строк в даты и время.

### **Практическое задание №4**

1. Ручное прохождение по итератору.
2. Создание новых итерационных паттернов с помощью генераторов.

3. Реализация протокола итератора.
4. Итерирование в обратном порядке.
5. Определение генератора с дополнительным состоянием.
6. Получение среза итератора.
7. Пропуск первой части итерируемого объекта.
8. Итерирование по всем возможным комбинациям и перестановкам.
9. Итерирование по парам «индекс–значение» последовательности.
10. Одновременное итерирование по нескольким последовательностям.
11. Итерирование по элементам, находящимся в отдельных контейнерах.
12. Создание каналов для обработки данных.
13. Превращение вложенной последовательности в плоскую.
14. Последовательное итерирование по слитым отсортированным итерируемым объектам.
15. Замена бесконечных циклов while итератором.

### **Практическое задание №5**

1. Чтение и запись текстовых данных.
2. Перенаправление вывода в файл.
3. Вывод с другим разделителем или символом конца строки.
4. Запись в файл, которого еще нет.
5. Выполнение операций ввода-вывода над строками.
6. Итерирование по записям фиксированного размера.
7. Оборачивание существующего дескриптора файла для использования в качестве объекта файла.
8. Создание временных файлов и каталогов.
9. Работа с последовательными портами.
10. Сериализация объектов Python.

### **Практическое задание №6**

1. Изменение строкового представления экземпляров.
2. Настройка строкового форматирования.
3. Создание объектов, поддерживающих протокол менеджера контекста.
4. Экономия памяти при создании большого количества экземпляров.
5. Инкапсуляция имен в классе.
6. Создание управляемых атрибутов.
7. Вызов метода родительского класса.
8. Расширение свойства в подклассе.
9. Создание нового типа атрибута класса или экземпляра.
10. Использование лениво вычисляемых свойств.
11. Упрощение инициализации структур данных.
12. Определение интерфейса или абстрактного базового класса.
13. Реализации модели данных или системы типов.
14. Реализация собственных контейнеров.
15. Делегирование доступа к атрибуту.
16. Определение более одного конструктора в классе.
17. Создание экземпляра без вызова `init`.
18. Расширение классов с помощью миксин (`mixins`).
19. Реализация объектов с состоянием или конечных автоматов.
20. Вызов метода объекта с передачей имени метода в строке.
21. Реализация шаблона проектирования «Посетитель».
22. Реализация шаблона «Посетитель» без рекурсии.
23. Управление памятью в циклических структурах данных.

24. Заставляем классы поддерживать операции сравнения.
25. Создание закешированных экземпляров.

### Практическое задание №7

1. Создание обертки для функции.
2. Сохранение метаданных функции при написании декораторов.
3. Снятие («разворачивание») декоратора.
4. Определение декораторов как классов.
5. Применение декораторов к методам класса и статическим методам.
6. Написание декораторов, которые добавляют аргументы обернутым функциям.
7. Использование декораторов для исправления определений классов.
8. Использование метакласса для управления созданием экземпляров.
9. Захват порядка определения атрибутов класса.
10. Определение метакласса, принимающего необязательные аргументы.
11. Принудительная установка аргументной сигнатуры#при использовании \*args и \*\*kwargs
12. Принуждение к использованию соглашений о кодировании в классах.
13. Программное определение классов.
14. Инициализация членов класса во время определения.
15. Реализация множественной диспетчеризации с помощью аннотаций функций.

### Требования к выполнению заданий практических работ № 1-7

Практические работы выполняются студентами в соответствии с заданием. Решения всех задач и пояснения к ним должны быть достаточно подробными. Решение заданий должно сопровождаться соответствующей аргументацией и выполняться самостоятельно.

#### Критерии оценивания практических работ

- 0-24 балла — «неудовлетворительно»
- 25-34 балла — «удовлетворительно»
- 35-44 балла — «хорошо»
- 45-50 баллов — «отлично»

#### Перечень лабораторных работ

##### Типовое задание для лабораторной работы

##### Лабораторная работа № 1

##### «Численное интегрирование»

**Цель работы:** отработать на практике использование циклов, функций, подготовку документации программных кодов.

**Требования к выполнению работы:** выполнение лабораторной работы предусматривает написание программы, реализующей численное интегрирование методом прямоугольников и методом трапеций. Отчёт о работе проводится в виде собеседования и заключается в демонстрации работы программы, объяснении принципов работы алгоритма и ответов на дополнительные вопросы.

**Критерии оценки:** для получения оценки «зачтено» необходимо показать высокий уровень владения теоретическим материалом, уметь объяснить принцип работы написанной программы, верно ответить на дополнительные вопросы.

**Задание:** написать программу, реализующую численное интегрирование методом прямоугольников и методом трапеций. Каждый метод оформляется отдельной функцией. Функция документируется в соответствии с принятыми отраслевыми стандартами. Проверить работу программы на контрольном примере

### **Типовое задание для лабораторной работы**

#### **Лабораторная работа № 2**

#### **«Нахождение корней уравнения»**

**Цель работы:** отработать на практике использование циклов, функций, подготовку документации программных кодов.

**Требования к выполнению работы:** выполнение лабораторной работы предусматривает написание программы, реализующей нахождение корней уравнения. Отчёт о работе проводится в виде собеседования и заключается в демонстрации работы программы, объяснении принципов работы алгоритма и ответов на дополнительные вопросы.

**Критерии оценки:** для получения оценки «зачтено» необходимо показать высокий уровень владения теоретическим материалом, уметь объяснить принцип работы написанной программы, верно ответить на дополнительные вопросы.

**Задание:** написать программу, реализующую нахождение корней уравнения. Метод оформляется отдельной функцией. Функция документируется в соответствии с принятыми отраслевыми стандартами. Проверить работу программы на контрольном примере.

### **Типовое задание для лабораторной работы**

#### **Лабораторная работа № 3**

#### **«Перемножение и транспонирование матриц»**

**Цель работы:** отработать на практике использование циклов, функций и списков, подготовку документации программных кодов.

**Требования к выполнению работы:** выполнение лабораторной работы предусматривает написание программы, реализующей перемножение и транспонирование матриц. Отчёт о работе проводится в виде собеседования и заключается в демонстрации работы программы, объяснении принципов работы алгоритма и ответов на дополнительные вопросы.

**Критерии оценки:** для получения оценки «зачтено» необходимо показать высокий уровень владения теоретическим материалом, уметь объяснить принцип работы написанной программы, верно ответить на дополнительные вопросы.

**Задание:** написать программу, реализующую перемножение и транспонирование матриц. Метод оформляется отдельной функцией. Функция документируется в соответствии с принятыми отраслевыми стандартами. Проверить работу программы на контрольном примере.

### **Типовое задание для лабораторной работы**

#### **Лабораторная работа № 4**

#### **«Интерполяция»**

**Цель работы:** отработать на практике использование циклов, функций, подготовку документации программных кодов.

**Требования к выполнению работы:** выполнение лабораторной работы предусматривает написание программы, реализующей различные варианты интерполяции. Отчёт о работе проводится в виде собеседования и заключается в демонстрации работы программы, объяснении принципов работы алгоритма и ответов на дополнительные вопросы.

**Критерии оценки:** для получения оценки «зачтено» необходимо показать высокий уровень владения теоретическим материалом, уметь объяснить принцип работы написанной программы, верно ответить на дополнительные вопросы.

**Задание:** написать программу, реализующую различные варианты интерполяции функций. Метод оформляется отдельной функцией. Функция документируется в соответствии с принятыми отраслевыми стандартами. Проверить работу программы на контрольном примере.

**Типовое задание для лабораторной работы  
Лабораторная работа № 5  
«Вычисление квадратного корня»**

**Цель работы:** отработать на практике использование циклов, функций, подготовку документации программных кодов.

**Требования к выполнению работы:** выполнение лабораторной работы предусматривает написание программы, реализующей нахождение квадратного корня методом Ньютона. Отчёт о работе проводится в виде собеседования и заключается в демонстрации работы программы, объяснении принципов работы алгоритма и ответов на дополнительные вопросы.

**Критерии оценки:** для получения оценки «зачтено» необходимо показать высокий уровень владения теоретическим материалом, уметь объяснить принцип работы написанной программы, верно ответить на дополнительные вопросы.

**Задание:** написать программу, реализующую нахождение квадратного корня методом Ньютона. Метод оформляется отдельной функцией. Функция документируется в соответствии с принятыми отраслевыми стандартами. Проверить работу программы на контрольном примере

**Типовое задание для лабораторной работы  
Лабораторная работа № 6  
«Организация ввода-вывода»**

**Цель работы:** отработать на практике использование циклов, функций, подготовку документации программных кодов.

**Требования к выполнению работы:** выполнение лабораторной работы предусматривает написание программы, реализующей ввод и вывод данных для работы с инструментами, реализованными в предыдущих лабораторных работах. Отчёт о работе проводится в виде собеседования и заключается в демонстрации работы

программы, объяснении принципов работы алгоритма и ответов на дополнительные вопросы.

**Критерии оценки:** для получения оценки «зачтено» необходимо показать высокий уровень владения теоретическим материалом, уметь объяснить принцип работы написанной программы, верно ответить на дополнительные вопросы.

**Задание:** написать программу, реализующую ввод и вывод данных для реализованных ранее функций. Проверить работу программы на контрольном примере

**Типовое задание для лабораторной работы**  
**Лабораторная работа № 7**  
**«Организация модульной структуры»**

**Цель работы:** отработать на практике использование модулей и пакетов, подготовку документации программных кодов.

**Требования к выполнению работы:** выполнение лабораторной работы предусматривает написание программы, реализующей модульную структуру программного продукта. Отчёт о работе проводится в виде собеседования и заключается в демонстрации работы программы, объяснении принципов работы алгоритма и ответов на дополнительные вопросы.

**Критерии оценки:** для получения оценки «зачтено» необходимо показать высокий уровень владения теоретическим материалом, уметь объяснить принцип работы написанной программы, верно ответить на дополнительные вопросы.

**Задание:** написать программу, реализующую математический пакет из ранее написанных инструментов. Проверить работу программы на контрольном примере.

**Типовое задание для лабораторной работы**  
**Лабораторная работа № 8**  
**«Обработка исключений»**

**Цель работы:** отработать на практике перехват исключений в программном коде на языке Python.

**Требования к выполнению работы:** выполнение лабораторной работы предусматривает оптимизацию ранее написанных программ. Отчёт о работе проводится в виде собеседования и заключается в демонстрации работы программы, объяснении принципов работы алгоритма и ответов на дополнительные вопросы.

**Критерии оценки:** для получения оценки «зачтено» необходимо показать высокий уровень владения теоретическим материалом, уметь объяснить принцип работы написанной программы, верно ответить на дополнительные вопросы.

**Задание:** добавьте в ранее написанные программы перехват возможных исключений. Проверить работу программы на контрольном примере.

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

Задания с выбором ответа

№	Задание	Варианты ответа	Верный ответ
1	Какой из приведенных методов не является методом строк?	1. S.find('pa') 2. S.replace('pa', 'x') 3. S.split(',') 4. 'spam'.join(strlist) 5. S.append('pa')	5
2	Какой из приведенных методов не является методом списков?	1. L.extend([5,6,7]) 2. L.isdigit() 3. L.reverse() 4. L.pop(i) 5. L.sort()	2

3	Какой из приведенных не является кортежем?	<ol style="list-style-type: none"> <li>1. T = (0,)</li> <li>2. T = (0, 'Ni', 1.2, 3)</li> <li>3. T = tuple('spam')</li> <li>4. T = (0)</li> <li>5. T = ('Bob', ('dev', 'mgr'))</li> </ol>	3
4	Какой из приведенных методов не является методом словарей?	<ol style="list-style-type: none"> <li>1. D.keys()</li> <li>2. D.values()</li> <li>3. D.remove()</li> <li>4. D.clear()</li> <li>5. D.popitem()</li> </ol>	3
5	Какая из типов не может быть ключем словаря?	<ol style="list-style-type: none"> <li>1. Число</li> <li>2. Строка</li> <li>3. Кортеж</li> <li>4. Список</li> </ol>	4
6	Какой из объектов относится к неизменяемому типу данных?	<ol style="list-style-type: none"> <li>1. res = list(map('ord', 'spam'))</li> <li>2. L = [3, 4.5, 6]</li> <li>3. S = 'abc'</li> <li>4. D = {'spam': 2, 'ham': 1}</li> <li>5. file = open('myfile.txt', 'w')</li> </ol>	3
7	Какой из приведенных не является числовым?	<ol style="list-style-type: none"> <li>1. Decimal('1.0')</li> <li>2. Fraction(1, 3)</li> <li>3. True</li> <li>4. 34-4j</li> <li>5. 0d9ff</li> </ol>	5
8	Какой из методов перегрузки операций отвечает за вызов функции?	<ol style="list-style-type: none"> <li>1. __call__</li> <li>2. __contains__</li> <li>3. __enter__</li> <li>4. __init__</li> <li>5. __repr__</li> </ol>	1
9	Какая из конструкций не относится к перехвату исключений?	<ol style="list-style-type: none"> <li>1. try/except</li> <li>2. try/finally</li> <li>3. raise</li> <li>4. assert</li> <li>5. @property</li> </ol>	5
10	Какая из конструкций не является строкой?	<ol style="list-style-type: none"> <li>1. S = r'\temp\spam'</li> <li>2. S = 's\np\tax00m'</li> <li>3. S = ''</li> <li>4. S = spam</li> </ol>	4

### Задания с кратким ответом

№	Задание	Верный ответ
1	Назовите наиболее распространенный метод перегрузки операций в Python?	__init__
2	Какой цикл в Python используется для прохода по итерируемому объекту?	for
3	Назовите встроенную функцию, которая выводит списки атрибутов, доступных в объектах.	dir()
4	Какой функцией выполняется перегрузка модулей в Python?	reload()
5	В каком каталоге хранятся файлы байт-кода в Python 3.2 и последующих версиях?	__pycache__

## Задания с развернутым ответом

**Задание 1.** Опишите общий формат записи цикла while.

Решение.

В своей самой сложной форме оператор while состоит из строки заголовка с выражением проверки, тела с одним или большим количеством оператором с отступами и необязательной части else, которая выполняется, если управление покидает цикл, а оператор break не встретился. Python продолжает оценивать выражение проверки в строке заголовка и выполняет операторы, вложенные в тело цикла, пока проверка не возвратит ложное значение:

```
while проверка:      # Проверка цикла
    операторы        # Тело цикла
else:                # Необязательная часть else
    операторы        # Выполняются, если не произведен выход из цикла с помощью break.
```

С учетом операторов break и continue общий формат цикла while выглядит следующим образом:

```
while проверка:      # Проверка цикла
    опера торы
    if проверка: break # Выход из цикла с пропуском else, если есть
    if проверка: continue # Переход на проверку в начале цикла
else:
    операторы        # Выполняется, если не было break
```

Критерии оценивания	Баллы
Имеется верная последовательность всех этапов решения, обоснованно получен верный ответ.	3
Получен неверный ответ из-за вычислительной ошибки, при этом имеется верная последовательность всех этапов решения.	2
Получен верный ответ, однако имеются пропуски одного или двух этапов решения ИЛИ Решение не завершено, однако верно выполнен хотя бы один из этапов решения.	1
Решение не соответствует ни одному из критериев, перечисленных выше.	0

**Задание 2.** Выполните возведение в квадрат каждого элемента в списке L = [1, 2, 3] с помощью ручного итерирования объекта через цикл while.

Решение.

Хотя итерационные инструменты Python вызывают функции iter() и next() автоматически, мы можем также использовать их для применения протокола итерации вручную.



```

>>> l = iter(L)                # Ручная итерация: то, что обычно делают циклы for
>>> while True:
...     try:                    # Оператор try перехватывает исключения
...         X = next(l)        # Или вызов l.__next__ в Python 3.X
...     except StopIteration:
...         break
...     print(X ** 2, end=' ')
>>> 149

```

Критерии оценивания	Баллы
Имеется верная последовательность всех этапов решения, обоснованно получен верный ответ.	3
Получен неверный ответ из-за вычислительной ошибки, при этом имеется верная последовательность всех этапов решения.	2
Получен верный ответ, однако имеются пропуски одного или двух этапов решения ИЛИ Решение не завершено, однако верно выполнен хотя бы один из этапов решения.	1
Решение не соответствует ни одному из критериев, перечисленных выше.	0

**Задание 3.** Напишите минимальный код для одновременного обхода по двум спискам L1 = [1,2,3,4] и L2 = [5,6,7,8] с попарным выводом их элементов. Объясните принцип работы кода.

Решение.

```

>>> for (x, y) in zip(L1, L2) :
...     print(x, y)
1 5
2 6
3 7
4 8

```

Здесь мы проходим по результату вызова zip, т.е. по парам элементов, извлеченных из двух списков. Цикл for снова применяет форму присваивания кортежей для распаковки каждого кортежа в результате zip. При первом проходе получается так, как если бы мы выполняли оператор присваивания (x, y) = (1, 5).

Совокупный эффект заключается в том, что мы просматриваем в цикле оба списка, L1 и L2. Похожего результата можно было бы добиться с помощью цикла while, который поддерживает индексацию вручную, но он потребовал бы большего объема набора.

Критерии оценивания	Баллы
Имеется верная последовательность всех этапов решения, обоснованно получен верный ответ.	3

Получен неверный ответ из-за вычислительной ошибки, при этом имеется верная последовательность всех этапов решения.	2
Получен верный ответ, однако имеются пропуски одного или двух этапов решения ИЛИ Решение не завершено, однако верно выполнен хотя бы один из этапов решения.	1
Решение не соответствует ни одному из критериев, перечисленных выше.	0

**Задание 4.** Опишите правила передачи аргументов в функции Python.

Решение.

1. Аргументы передаются путем автоматического присваивания объектов именам локальных переменных. Аргументы функций, т.е. ссылки на (возможно) разделяемые объекты, отправленные вызывающим кодом, представляют собой присваивания. Поскольку ссылки реализованы в виде указателей, все аргументы в действительности передаются через указатели. Объекты, передаваемые как аргументы, никогда автоматически не копируются.
2. Присваивание именам аргументов внутри функции не затрагивает вызывающий код. Когда функция выполняется, имена аргументов в заголовке функции становятся новыми локальными именами в области видимости функции. Никакого совмещения имен аргументов функции и имен переменных в области видимости вызывающего кода не происходит.
3. Модификация внутри функции аргумента, являющегося изменяемым объектом, может затронуть вызывающий код. С другой стороны, так как аргументам просто присваиваются передаваемые объекты, в функциях можно модифицировать переданные изменяемые объекты на месте, в результате оказывая влияние на вызывающий код. Изменяемые аргументы могут служить входными и выходными данными для функций.
4. Неизменяемые аргументы фактически передаются “по значению”. Объекты, подобные целым числам и строкам, передаются по ссылке на объекты, а не путем копирования, но из-за того, что модифицировать на месте неизменяемые объекты невозможно, эффект во многом похож на создание копий.
5. Изменяемые аргументы фактически передаются “по указателю”. Объекты вроде списков и словарей также передаются по ссылке на объекты, что аналогично способу передачи массивов как указателей в языке C — изменяемые объекты можно модифицировать на месте в функции почти как массивы в C.

Критерии оценивания	Баллы
Имеется верная последовательность всех этапов решения, обоснованно получен верный ответ.	3
Получен неверный ответ из-за вычислительной ошибки, при этом имеется верная последовательность всех этапов решения.	2
Получен верный ответ, однако имеются пропуски одного или двух этапов	1

решения ИЛИ Решение не завершено, однако верно выполнен хотя бы один из этапов решения.	
Решение не соответствует ни одному из критериев, перечисленных выше.	0

**Задание 5.** Создайте класс Number, который осуществляет перегрузку операции `__sub__`, превращая вычитание в сложение.

Решение.

Класс Number предоставляет метод для перехвата создания экземпляра (`__init__`), а также метод для отлавливания выражений вычитания (`__sub__`) и преобразования их в сложение. Специальные методы подобного рода являются привязками, которые дают возможность соединяться со встроенными операциями:

```
class Number:
    def __init__(self, start):          # Для Number(start)
        self.data = start
    def __sub__(self, other) :        # Для экземпляр - other
        return Number(self.data + other) # Результатом будет новый экземпляр
```

Критерии оценивания	Баллы
Имеется верная последовательность всех этапов решения, обоснованно получен верный ответ.	3
Получен неверный ответ из-за вычислительной ошибки, при этом имеется верная последовательность всех этапов решения.	2
Получен верный ответ, однако имеются пропуски одного или двух этапов решения ИЛИ Решение не завершено, однако верно выполнен хотя бы один из этапов решения.	1
Решение не соответствует ни одному из критериев, перечисленных выше.	0

## 20.2. Промежуточная аттестация

Промежуточная аттестация по дисциплине осуществляется с помощью следующих оценочных средств: перечень вопросов к зачёту.

### 20.2.1 Перечень вопросов к зачёту

Раздел 1. Типы и операции.

1. Основные типы объектов Python.
2. Числовые типы.
3. Операции выражений Python и старшинство.
4. Множества. Включения множеств в Python.
5. Динамическая типизация.
6. Строковый тип.
7. Методы строк.
8. Форматирования строк.

- 9. Списки в Python. Списковые литералы и операции.
- 10. Словари. Словарные литералы и операции.
- 11. Методы словарей.
- 12. Кorteжи.
- 13. Файлы. Файловые операции.

Раздел 2. Операторы и синтаксис.

- 14. Формы оператора присваивания. Проверки if.
- 15. Циклы while и for.
- 16. Документация.

Раздел 3. Функции.

- 17. Основы функций.
- 18. Области видимости.
- 19. Аргументы функций. Вызов функций.
- 20. Списковые включения.

Раздел 4. Модули и пакеты.

- 21. Модули. Операции импорта.
- 22. Пакеты модулей и пакетов пространств имен.
- 23. Измерение времени выполнения.

Раздел 5. Классы и объектно-ориентированное программирование.

- 24. Объектно-ориентированное программирование. Основные концепции.
- 25. Основы написания классов.
- 26. Детали реализации классов.
- 27. Перегрузка операций.
- 28. Декораторы.
- 29. Метаклассы.

Раздел 6. Исключения.

- 30. Оператор try/except/finally.
- 31. Оператор raise и assert.
- 32. Диспетчеры контекстов with/as.
- 33. Объекты исключений.

**Требования к студентам при проведении промежуточной аттестации**

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

**Требования к студентам при проведении промежуточной аттестации**

Критерии оценивания компетенций	Уровень сформированности компетенций	Шкала оценок
Полное соответствие ответа обучающегося всем перечисленным критериям. Обучающийся демонстрирует высокий уровень владения материалом, ориентируется в предметной области, верно отвечает на все дополнительные вопросы.	Повышенный уровень	Отлично
Ответ на контрольно-измерительный материал не соответствует одному или двум из перечисленных	Базовый уровень	Хорошо

показателей, но обучающийся дает правильные ответы на дополнительные вопросы. Допускаются ошибки при воспроизведении части теоретических положений.		
Ответ на контрольно-измерительный материал не соответствует любым трём из перечисленных показателей, обучающийся дает неполные ответы на дополнительные вопросы. Сформированные знания основных понятий, определений и теорем, изучаемых в курсе, не всегда полное их понимание с затруднениями при воспроизведении.	Пороговый уровень	Удовлетворительно
Ответ на контрольно-измерительный материал не соответствует любым четырём из перечисленных показателей. Обучающийся демонстрирует отрывочные знания (либо их отсутствие) основных понятий, определений и теорем, используемых в курсе.	–	Неудовлетворительно