

Федеральное государственное образовательное бюджетное учреждение
высшего образования

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ
РОССИЙСКОЙ ФЕДЕРАЦИИ»
(Финансовый университет)**

**Кафедра информационных технологий
Факультет информационных технологий и анализа больших данных**

Документ подписан усиленной неквалифицированной электронной подписью
Организация: Финансовый университет при Правительстве РФ
Утверждено: Проректор по учебной и методической работе Е.А. Каменева
Сертификат: QLwIQIMOANuK1VU14ViZZypLVokVU/r7
Дата: 25.03.2025 г.

Р. И. Горохова

Алгоритмы и структуры данных в языке Python

Рабочая программа дисциплины

для студентов, обучающихся по направлению подготовки

09.03.04 - Программная инженерия

Образовательная программа:

Разработка и внедрение информационно - аналитических систем

Профиль

Разработка и внедрение информационно - аналитических систем

Рекомендовано

*Факультет информационных технологий и анализа больших данных
(протокол №55 от 20.05.2025)*

Одобрено

*Кафедра информационных технологий
(протокол №04 от 19.05.2025)*

Москва 2025

Содержание

Наименование разделов РПД		Стр.
1.	Наименование дисциплины	5
2.	Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине	5
3.	Место дисциплины в структуре образовательной программы	7
4.	Объем дисциплины(модуля) в зачетных единицах и в академических часах с выделением объема аудиторной (лекции, семинары) и самостоятельной работы обучающихся	8
5.	Содержание дисциплины, структурированное по темам (разделам) дисциплины с указанием их объемов (в академических часах) и видов учебных занятий	8
5.1.	Содержание дисциплины	8
5.2.	Учебно-тематический план	13
5.3.	Содержание семинаров, практических занятий	15
6.	Перечень учебно-методического обеспечения для	19

	самостоятельной работы обучающихся по дисциплине	
6.1.	Перечень вопросов, отводимых на самостоятельное освоение дисциплины, формы внеаудиторной самостоятельной работы	20
6.2.	Перечень вопросов, заданий, тем для подготовки к текущему контролю	23
7.	Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине	26
8.	Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины	38
9.	Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины	40
10.	Методические указания для обучающихся по освоению дисциплины	40
11.	Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень необходимого программного обеспечения и информационных	41

	справочных систем	
12.	Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине	42

1. Наименование дисциплины

«Алгоритмы и структуры данных в языке Python».

2. Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине

Код компетенции	Наименование компетенции	Индикатор достижения компетенции	Результаты обучения
ОПК-6	Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов	Разрабатывает алгоритмы решения простых информационных задач и выражает их на языке программирования	Знать: объектно-ориентированный язык программирования Python на уровне знания синтаксиса и семантики, основ стандартной библиотеки Уметь: определять на уровне знания синтаксиса и семантику, стандартные библиотеки языка Python, необходимые для решения прикладных задач
		Анализирует алгоритмы в части производительности, оптимальности, вырабатывает рекомендации для	Знать: технологии прикладного программирования, инструментальные средства программирования

		оптимизации алгоритмов программ	Уметь: разрабатывать программы решения задач с использованием прикладного программирования, включая среды высокоуровневого программирования
		Проводит ручное и автоматизированное тестирование программных продуктов по методам черного и белого ящика, составлять набор тестовых случаев	Знать: особенности использования технологии Web-программирования Уметь: разрабатывать программный код, ориентироваться в существующем коде, применять знание общепринятых соглашений и политик в области оформления кода
ОПК-7	Способен применять в практической деятельности основные концепции, принципы, теории и факты, связанные с информатикой	Демонстрирует знания основ теории информации и алгоритмов, основных элементарных алгоритмов и структуры данных	Знать: основы теории информации и алгоритмов, основные элементарные алгоритмы и структуры данных, процессы сбора, обработки и интерпретации информации Уметь:

			описывать состав и структуру требуемых данных и информации
		Применяет простые алгоритмы и структуры данных к решению поставленной задачи, проводит выбор наиболее оптимальных методов	Знать: простые алгоритмы и структуры данных, оптимальные методы решения задач Уметь: применять простые алгоритмы и структуры данных к решению поставленной задачи, проводить выбор наиболее оптимальных методов
		Проводит подробный количественных анализ реализованной программной системы с точки зрения оптимальности применяемых алгоритмических решений	Знать: основные методы количественного анализа программных систем Уметь: выполнять количественных анализ реализованной программной системы с точки зрения оптимальности применяемых алгоритмических решений

3. Место дисциплины в структуре образовательной программы

Дисциплина «Алгоритмы и структуры данных в языке Python» относится к «Общепрофессиональному циклу»

4. Объем дисциплины(модуля) в зачетных единицах и в академических часах с выделением объема аудиторной (лекции, семинары) и самостоятельной работы обучающихся

Очная форма обучения

Вид учебной работы по дисциплине	Всего (в з/е и часах)	Семестр 2 (в часах)	Семестр 1 (в часах)
Общая трудоемкость дисциплины	8/288	144	144
Контактная работа - Аудиторные занятия	100	50	50
<i>Лекции</i>	32	16	16
<i>Семинары, практические занятия</i>	68	34	34
Самостоятельная работа	188	94	94
Вид текущего контроля	Контрольная работа; Контрольная работа;	Контрольная работа	Контрольная работа
Вид промежуточной аттестации	Экзамен; Экзамен;	Экзамен	Экзамен

5. Содержание дисциплины, структурированное по темам (разделам) дисциплины с указанием их объемов (в академических часах) и видов учебных занятий

5.1. Содержание дисциплины

Тема 1. Введение в программирование на Python

Общая информация о языке Python . История языка программирования, его связь с другими языками программирования, распространенность Python и основные сферы

его применения. Знакомство с первыми примерами кода на Python. Философия Python.

Базовая информация о языке Python . Основные типы данных. Основные числовые типы данных и операции над ними. Математические операции над числовыми типами данных. Преобразование типов данных. Переменные и специфика их объявления. Статическая и динамическая типизация. Работа с переменными. Управление памятью и сборка мусора в Python . Именованые переменных.

Работа со строками: создание строк, специальные символы. Индексирование строк, получение срезов строк. Основные функции для работы со строками. Вывод на экран (работа с функцией print) и форматирование строк. Различные подходы к форматированию строк, форматирование с помощью f-строк. Расширенное форматирование в Python.

Тема 2. Управляющие конструкции, списки и кортежи

Управляющие конструкции в Python . Булев тип: объявление и операции. Операции сравнения в Python . Условные операторы в Python . Реализация задачи case в Python .

Циклы в Python: while, for. Специфика циклов в Python . Функции range и enumerate и их использование в циклах.

Списки и кортежи в Python . Специфика списков и их отличие от массивов. Создание списка, оперирование вложенными списками, копирование списков, операции над списками: индексация и срезы; изменение списка; поиск, сортировка и обход; изменение списка. Кортежи в Python : синтаксис, специфика использования.

Тема 3. Словари, множества и выражения-генераторы

Словари Python . Словари: семантика, синтаксис создания, операции над словарями, перебор элементов словаря.

Множества в Python . Множества: семантика, синтаксис создания, операции над словарями, перебор элементов словаря. Специфика операций с множествами в Python .

Выражения-генераторы в Python . Выражения-генераторы для списков: семантика и синтаксис. Пример: задача приведения

списка к "плоскому" виду. Выражения-генераторы для множеств и словарей. Кейсы использования и производительность решений с использованием выражений-генераторов.

Тема 4. Функции

Функции в Python : общая семантика. Создание функции и ее вызов. Расположение определений функций. Анонимные функции в Python . Необязательные параметры функций и сопоставление по ключам. Возвращение нескольких значений из функции. Распаковка и запаковка параметров функции. Аннотации и документирование функций. Глобальные и локальные переменные.

Тема 5. Работа с файлами и обработка исключительных ситуаций

Обработка исключений в Python : кейсы для использования. Инструкция `try ... except ... else ... finally`. Классы встроенных исключений. Создание пользовательских исключений. Инструкция `assert`.

Работа с файлами в Python . Концепция файла в современных ОС и языках программирования. Операции с файлами: открытие/закрытие файла, чтение и записи и другие методы для работы с файлами. Инструкция `with ... as` и ее использование для файлов.

Сохранение объектов в файл с помощью модуля `pickle` и `shelve` . Модуль `CSV` .

Тема 6. Модули и пакеты

Модули и пакеты в Python : подход к структурированию программного кода с помощью модулей и пакетов. Синтаксис импортирования в Python . Создание и работа с пакетами в Python . Повторная загрузка модулей.

Написание и запуск скриптов на Python. Установка модулей из глобального репозитория.

Тема 7. Продвинутое коллекции

Модули стандартной библиотеки Python , предоставляющие дополнительный функционал по работе с коллекциями объектов.

Класс Frozen set. Модуль collections. Класс Counter , класс defaultdict , класс OrderedDict , класс namedtuple . Модуль enum .

Тема 8. Обзор современных языков программирования

История и эволюция языков программирования. Распространенность современных языков программирования. Классификация языков программирования. Парадигмы программирования. Специализация языков программирования. Место Python в современном ландшафте языков программирования.

Характеристики языков программирования: способ передачи параметров в функцию; способ управления динамической памятью; виды типизации переменных.

Тема 9. Введение в объектно-ориентированное программирование

Предпосылки и история появления ООП. Объекты и классы в ООП. Принципы и основные механизмы ООП. Логика работы абстракции, инкапсуляции, наследования и полиморфизма.

Python как объектно-ориентированный язык программирования. Базовые возможности ООП в Python : с оздание классов и объектов; наследование и полиморфизм; функция super(); проверка принадлежности к классу. Базовые типы в Python.

Тема 10. Объектно-ориентированное программирование в Python

Методы классов и статические переменные и методы в Python . Управление доступом к атрибутам класса в Python . Динамические операции с атрибутами и интроспекция в Python . Использование специальных методов для расширенного функционала пользовательских классов. Кейс построения иерархии классов.

Тема 11. Введение в функциональное программирование

Парадигмы и идиомы программирования, общая концепция функционального программирования. Функциональные языки программирования.

Функции "граждане первого класса", функции высшего порядка, замыкания, функции без побочных эффектов, рекурсия, хвостовая

рекурсия. Неизменяемые структуры данных. Идиомы, распространенные в функциональных языках программирования: итераторы, последовательности, ленивые вычисления, сопоставление с образцом, монады.

Элементы функционального программирования в Python : функции – граждане первого класса; глобальные и локальные переменные в Python ; вложенные функции и замыкания в Python .

Декораторы в Python : использование и создание собственных декораторов.

Тема 12. Функциональное программирование в Python

Подход: map , filter , reduce . Реализация функций map , filter , reduce в Python .

Итераторы в Python , итерируемый тип данных. Модуль itertools.

Функции-генераторы и выражения-генераторы в Python .

Тема 13. Структуры данных: массивы, стеки, очереди, списки

Введение в анализ сложности алгоритмов.

Массивы и их отличие от списков в Python . Динамические массивы, сложность операций работы с динамическими массивами.

Стек, операции со стеком. Реализации стека. Очередь, операции с очередью. Реализация очереди. Связные списки, варианты связанных списков.

Тема 14. Алгоритмы поиска и сортировки

Поиск в списках/массивах, бинарный поиск. Сортировка и ее использование в прикладных задачах.

Простые методы сортировки: обменные сортировки (с различными вариациями); сортировка выбором (извлечением); сортировка включением (вставками).

Эффективные методы сортировки: быстрая сортировка; сортировка Шелла; сортировка слиянием. Сравнение различных сортировок.

Тема 15. Структуры данных: деревья

Деревья, бинарные деревья. Использование бинарных деревьев в прикладных задачах: представление выражения (предложения) в виде дерева. Обход бинарных деревьев.

Двоичное дерево поиска. Двоичные кучи и очереди с приоритетом.

Тема 16. Хеш-таблицы

Абстрактный тип данных – ассоциативный массив. Таблица с прямой адресацией. Хеш-таблица, хеш-функция: метод деления; метод MAD. Полиномиальная хеш-функция.

Функция hash в Python.

Методы разрешения коллизий.

5.2. Учебно-тематический план

Очная форма обучения

№ п/п	Наименование тем(разделов) дисциплины	Трудоемкость в часах				
		Всего	Контактная работа - Аудиторная работа			Самостоятельная работа
			Общая, в т.ч.:	Лекции	Семинары, практические занятия	
1	Введение в программирование на Python	16	6	2	4	10
2	Управляющие конструкции, списки и кортежи	18	6	2	4	12
3	Словари, множества и выраже	20	8	2	6	12

	ния-генераторы					
4	Функции	18	6	2	4	12
5	Работа с файлами и обработка исключительных ситуаций	20	8	2	6	12
6	Модули и пакеты	18	6	2	4	12
7	Продвинутые коллекции	18	6	2	4	12
8	Обзор современных языков программирования	14	4	2	2	10
9	Введение в объектно-ориентированное программирование	18	6	2	4	12
10	Объектно-ориентированное программирование в Python	20	8	2	6	12
11	Введение	18	6	2	4	12

	е в функцио нальное програм мирован ие					
12	Функцио нальное програм мирован ие в Python	18	6	2	4	12
13	Структу ры данных: массивы , стеки, очереди, списки	18	6	2	4	12
14	Алгорит мы поиска и сортиро вки	18	6	2	4	12
15	Структу ры данных: деревья	18	6	2	4	12
16	Хеш- таблицы	18	6	2	4	12
	Итого	288	100	32	68	188

5.3. Содержание семинаров, практических занятий

Наименование тем (разделов) дисциплины	Перечень вопросов для обсуждения на семинарах, практических занятиях	Формы проведения занятий
Введение в программирование на Python	Установка Python, установка дистрибутива Anaconda. Работа в интерактивном режиме интерпретатора. Интерактивная оболочка IPython	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).

	notebook: принципы работы и применения.	
Управляющие конструкции, списки и кортежи	Базовые числовые типы, строки, списки, словари, переменные, базовые операторы.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Словари, множества и выражения-генераторы	Множества в Python. Множества: семантика, синтаксис создания, операции над словарями, перебор элементов словаря. Специфика операций с множествами в Python. Выражения-генераторы в Python. Выражения-генераторы для списков: семантика и синтаксис.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Функции	Создание функций, область видимости переменной, передача аргументов в функцию. Лямбда-функции.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Работа с файлами и обработка исключительных ситуаций	Исключения. Инструкция try...except...else...finally. Классы встроенных исключений. Создание пользовательских исключений. Инструкция assert. Работа с файлами в Python, операции с файлами: открытие/закрытие файла, чтение и записи и другие	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).

	методы для работы с файлами. Инструкция with ... as и ее использование для файлов.	
Модули и пакеты	Устройство модулей и пакетов, инструкции import и from. Создание собственных модулей и пакетов.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Продвинутые коллекции	Модули стандартной библиотеки Python, предоставляющие дополнительный функционал по работе с коллекциями объектов. Класс Frozen set. Модуль collections.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Обзор современных языков программирования	Парадигмы программирования. Специализация языков программирования. Место Python в современном ландшафте языков программирования. Характеристики языков программирования: способ передачи параметров в функцию; способ управления динамической памятью.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Введение в объектно-ориентированное программирование	Базовые возможности ООП в Python: создание классов и объектов; наследование и полиморфизм; функция super().	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).

Объектно-ориентированное программирование в Python	Методы классов и статические переменные и методы в Python. Управление доступом к атрибутам класса в Python. Динамические операции с атрибутами и интроспекция в Python. Использование специальных методов для расширенного функционала пользовательских классов. Кейс построения иерархии классов.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Введение в функциональное программирование	Элементы функционального программирования в Python: функции – граждане первого класса; глобальные и локальные переменные в Python; вложенные функции и замыкания в Python. Декораторы в Python: использование и создание собственных декораторов.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Функциональное программирование в Python	Реализация функций map, filter, reduce в Python. Итераторы в Python, итерируемый тип данных. Модуль itertools.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Структуры данных: массивы, стеки, очереди, списки	Массивы и их отличие от списков в Python. Динамические массивы, сложность операций работы с динамическими массивами. Стек, операции со стеком. Реализации	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).

	стека. Очередь, операции с очередью. Реализация очереди. Связные списки, варианты связанных списков.	
Алгоритмы поиска и сортировки	Реализация на Python простых методов сортировки: обменные сортировки (с различными вариациями); сортировка выбором (извлечением); сортировка включением (вставками). Реализация на Python эффективных методов сортировки: быстрая сортировка; сортировка Шелла; сортировка слиянием.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Структуры данных: деревья	Реализация на Python деревьев, бинарных деревьев. Использование бинарных деревьев в прикладных задачах: представление выражения (предложения) в виде дерева.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).
Хеш-таблицы	Работа с функцией hash в Python. Использование специализированных библиотек для работы с хэш-функциями.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии).

6. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине

6.1. Перечень вопросов, отводимых на самостоятельное освоение дисциплины, формы внеаудиторной самостоятельной работы

Наименование тем (разделов) дисциплины	Перечень вопросов, отводимых на самостоятельное освоение	Формы внеаудиторной самостоятельной работы
Введение в программирование на Python	Среда программирования. Изучение и подготовки к использованию документации языка программирования Python.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Управляющие конструкции, списки и кортежи	Оперирование вложенными списками, копирование списков, некоторые операции над списками.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Словари, множества и выражения-генераторы	Выражения-генераторы в Python. Выражения-генераторы для словарей и множеств: семантика, синтаксис и практическое использование.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Функции	Возвращение нескольких значений из функции. Распаковка и запаковка параметров функции.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Работа с файлами и обработка исключительных	Работа с текстовыми, бинарными и табличными файлами.	Индивидуальное выполнение практических заданий

ситуаций	Сохранение объектов в файл с помощью модуля pickle и shelve . Модуль CSV .	с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Модули и пакеты	Написание и запуск скриптов на Python. Установка модулей из глобального репозитория.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Продвинутые коллекции	Класс Counter , класс defaultdict , класс OrderedDict , класс namedtuple . Модуль enum .	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Обзор современных языков программирования	Виды типизации переменных в различных современных языках программирования.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Введение в объектно-ориентированное программирование	Базовые типы в Python: взгляд с точки зрения ООП. Методы базовых типов.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Объектно-ориентированное программирование в Python	Принципы ООП: абстракция, инкапсуляция, наследование, полиморфизм. Проверка принадлежности к	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки

	классу и интроспекция в Python.	программного кода на языке Python.
Введение в функциональное программирование	Функции-генераторы и выражения-генераторы в Python. Реализация декораторов с параметрами в Python.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Функциональное программирование в Python	Функции-генераторы и выражения-генераторы в Python. Реализация декораторов с параметрами в Python.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Структуры данных: массивы, стеки, очереди, списки	Структуры данных: массивы, стеки, очереди, списки. Реализация различных вариантов связанных списков на Python.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Алгоритмы поиска и сортировки	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Структуры данных: деревья	Двоичный поиск. Реализация обхода бинарных деревьев.	Индивидуальное выполнение практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
Хеш-таблицы	Использование специализированных	Индивидуальное выполнение

	библиотек для работы с хэш-функциями: проверка хэш-конфликтов для различных наборов данных.	практических заданий с использованием Jupyter Notebook или другой среды разработки программного кода на языке Python.
--	---	---

6.2. Перечень вопросов, заданий, тем для подготовки к текущему контролю

Примерные вопросы к контрольной работе (1 семестр)

1. Операции над основными числовыми типами данных.
2. Операции над булевыми переменными.
3. Динамическая типизация в Python.
4. Преобразование типов в Python.
5. Создание строк в Python.
6. Организация и пример цикла while в Python.
7. Организация и пример цикла for в Python.
8. Операции над словарями в Python.
9. Выражения-генераторы для списков в Python.
10. Необязательные параметры функций в Python.
11. Заpackовка и распаковка параметров в Python.
12. Аннотации и документирование функций.
13. Создание объектов в Python.
14. Управление доступом к атрибутам класса в Python.
15. Выполнение интроспекции в Python.
16. Замыкания в Python.
17. Использование декораторов в Python.
18. Создание собственных декораторов в Python.

19. Компилятор и интерпретатор. Достоинства и недостатки.
20. Назовите и дайте краткую характеристику основных классов языков программирования.
21. Встроенные числовые типы языка Python.
22. Списки. Создание, основные операции.
23. Основные методы списка.
24. Кортежи. Создание, основные методы и операции.
25. Словари. Создание, основные операции.
26. Методы для работы со словарями.
27. Множества. Создание, основные методы и операции.
28. Переменные. Правила именования переменных.
29. Динамическая типизация.
30. Операторы сравнения и логические операторы.
31. Инструкция if...else.
32. Инструкция цикла while.
33. Инструкция цикла for.
34. Создание и вызов функции.
35. Передача аргументов функцию.
36. Функции-генераторы.
37. Лямбда-функции.
38. Модули. Инструкции import и from.

Примерные вопросы к контрольной работе (2 семестр)

1. Базовые принципы объектно-ориентированного программирования.
2. Класс, метод класса, атрибут класса. Определение класса и создание экземпляра класса.

3. Конструктор и деструктор.
4. Наследование.
5. Абстрактные методы класса.
6. Статические методы класса.
7. Свойства класса.
8. Исключения. Обработка исключений.
9. Пользовательские исключения.
10. Вложенные функции и замыкания, специфика реализации в Python.
11. Функции высшего порядка и декораторы в Python.
12. Реализация map/filter/reduce в Python и пример их использования.
13. Итераторы в Python.
14. Встроенные функции для работы с итераторами и возможности модуля itertools.
15. Специфика массивов, как структур данных.
16. Абстрактная структура данных стек и очередь: базовые и расширенные операции, их сложность.
17. Реализация основных операций в очереди на базе массива и связанного списка.
18. Связанные списки: однонаправленные и двунаправленные – принцип реализации.
19. Алгоритм обменной сортировки.
20. Алгоритм сортировки выбором.
21. Алгоритм сортировки вставками.
22. Алгоритм быстрого поиска в отсортированном массиве.
23. Алгоритм сортировки Шелла.
24. Алгоритм быстрой сортировки.

25. Алгоритм сортировки слиянием.
26. Структуры данных: деревья
27. Реализация двоичных деревьев в виде связанных объектов. Различные реализации рекурсивного обхода двоичных деревьев.
28. Двоичная куча, реализации основных операций.
29. Абстрактный тип данных - ассоциативный массив и принцип его реализации на основе хэш-таблиц и хэш-функций.
30. Хэш - функции multiply-add-and-divide. Принцип работы хэш - функции multiply-add-and-divide.
31. Полиномиальная хэш-функция.

Критерии балльной оценки различных форм текущего контроля успеваемости содержатся в соответствующих методических рекомендациях Кафедры информационных технологий.

7. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине

Перечень компетенций с указанием индикаторов их достижения в процессе освоения образовательной программы содержится в разделе 2. *Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине.*

Типовые контрольные задания или иные материалы, необходимые для оценки индикаторов достижения компетенций, умений и знаний

ОПК-6 Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов

1) Разрабатывает алгоритмы решения простых информационных задач и выражает их на языке программирования

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: объектно-ориентированный язык программирования Python на уровне знания синтаксиса и семантики, основ стандартной библиотеки

Уметь: определять на уровне знания синтаксис и семантику, стандартные библиотеки языка Python, необходимые для решения прикладных задач

Типовые контрольные задания

1. Напишите программу на Python, обрабатывающую текстовую строку так, чтобы в нем все заглавные буквы преобразовать в строчные.
2. Напишите скрипт на Python обрабатывающий текстовый файл и получение на его основе словаря, ключами являются слова текста, а значениями количество встречаемости слов. Сохраните в файле CSV .

2) Анализирует алгоритмы в части производительности, оптимальности, вырабатывает рекомендации для оптимизации алгоритмов программ

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: технологии прикладного программирования, инструментальные средства программирования

Уметь: разрабатывать программы решения задач с использованием прикладного программирования, включая среды высокоуровневого программирования

Типовые контрольные задания

1. Используя программу Jupyter Notebook составьте код на Python, определяющий количество точек на плоскости, находящихся на заданном расстоянии от начала координат. Используйте библиотеку math .
2. Выберите библиотеку Python для работы с массивами. Выполните программный код бинарного поиска данных.

3) Проводит ручное и автоматизированное тестирование программных продуктов по методам черного и белого ящика, составлять набор тестовых случаев

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: особенности использования технологии Web-программирования

Уметь: разрабатывать программный код, ориентироваться в существующем коде, применять знание общепринятых соглашений и политик в области оформления кода

Типовые контрольные задания

1. Выберите библиотеку Python для создания и обработки деревьев данных. Реализуйте программный код вычисления различных арифметических операций, выбор которых осуществляется с помощью разработанного меню.

2. Реализуйте программу на Python которой в качестве аргумента командной строки передается имя CSV-файла, в первом столбце находятся числа, которые необходимо отсортировать. Программа создает новый файл, в котором первый столбец отсортирован.

ОПК-7 Способен применять в практической деятельности основные концепции, принципы, теории и факты, связанные с информатикой

1) Демонстрирует знания основ теории информации и алгоритмов, основных элементарных алгоритмов и структуры данных

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: основы теории информации и алгоритмов, основные элементарные алгоритмы и структуры данных, процессы сбора, обработки и интерпретации информации

Уметь: описывать состав и структуру требуемых данных и информации

Типовые контрольные задания

1. Напишите скрипт на Python обрабатывающий текстовый файл так, что в нем все заглавные буквы преобразуются в строчные.

2. Напишите скрипт на Python обрабатывающий текстовый файл и получение на его основе словаря, ключами являются слова текста, а

значениями количество встречаемости слов. Сохраните в файле CSV .

2) Применяет простые алгоритмы и структуры данных к решению поставленной задачи, проводит выбор наиболее оптимальных методов

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: простые алгоритмы и структуры данных, оптимальные методы решения задач

Уметь: применять простые алгоритмы и структуры данных к решению поставленной задачи, проводить выбор наиболее оптимальных методов

Типовые контрольные задания

1. Установите пакет Anaconda запустите программу Jupyter Notebook составьте тестовый код на Python. Выберите библиотеку Python для разработки системы управления складом малого предприятия.

2. Реализуйте программу на Python, которая реализует систему управления складом малого предприятия.

3) Проводит подробный количественных анализ реализованной программной системы с точки зрения оптимальности применяемых алгоритмических решений

Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции

Знать: основные методы количественного анализа программных систем

Уметь: выполнять количественных анализ реализованной программной системы с точки зрения оптимальности применяемых алгоритмических решений

Типовые контрольные задания

1. Выполните обработку текстового файла, в котором зафиксированы факты продаж, необходимого для подсчета суммарных продаж по различным географическим подразделениям фирмы. На основе полученных данных составьте CSV файл, хранящий полученные результаты.

2. Опишите структуру CSV файла фиксирующего факты продаж, необходимого для подсчета суммарных продаж по различным географическим подразделениям фирмы.

Примеры практико-ориентированных заданий

1 семестр

1. В строке содержащей последовательность слов, разделенных запятыми удалить все нечетные слова. Ответ представить в виде строки. Пример: строка 'SIX,SEVEN,EIGHT,NINE,TEN' будет преобразована в: 'SIX,EIGHT,TEN'.

2. Из списка списков элементами которого являются текстовые символы собрать строку, в которой вложенные списки объединены в слова, а слова через запятую объединены в строку. Пример список вида [['E', 'e', 'n', 'y'], ['m', 'e', 'e', 'n', 'y'], ['m', 'i', 'n', 'e', 'y'], ['m', 'o', 'e']] будет преобразован в строку 'Eeny,meeny,miney,moe'.3. Используя генератор словарей (и не используя код вне него) инвертировать словарь, т.е. сделать ключи словаря, его значениями и наоборот. Значения, которые в исходном словаре повторяются не добавлять в итоговый словарь. Пример: {'a':1, 'b':3, 'c':4, 'd':3} -> {1:'a', 4:'c'}.

3. Задана строка, в которой через запятую перечислены имена людей (с заглавной буквы) и их текущие занятия (со строчной буквы) в произвольном порядке (например, "Иван ест, поет Оля" и т.д.). С помощью генераторов создать словарь, в котором ключами будут имена, а значениями – занятия. Решить задачу в одну строку. Например: "Маша гуляет, Коля работает, дома Ваня, закупается Женя" представить в виде {'Ваня': 'дома', 'Женя': 'закупается', 'Коля': 'работает', 'Маша': 'гуляет'}.

4. Написать скрипт, который заменяет в текстовом файле все слова из определенного перечня на определенные для этих слов слова-заменители. Скрипт принимает через командную строку 2 параметра: первый параметр - имя файла из которого берется текст (файл имеет кодировку UTF8; нет переносов слов на другую строку); второй параметр - имя файла, являющегося файлом CSV (с разделителем ;) содержащего перечень слов для замены и

соответствующих им заменителей (файл не имеет заголовка столбцов; в первом столбце слова подлежащие замене, во втором слова-заменители) (файл имеет кодировку UTF8). Результат замены слов сохраняется в новом файле с именем совпадающим с исходным файлом с текстом, но имеющим префикс подчеркивание (_). Скрипт должен иметь вид `replace_Ivanov.py` (вместо `Ivanov` - ваша фамилия). Функции связанные с обработкой файлов и заменой в тексте должны находиться в отдельном модуле `modul_Ivanov.py` (вместо `Ivanov` - ваша фамилия).

5. Реализовать функцию `summmulate` для расчета накопленных двойных сумм (квадратов произведений). Функция принимает одно или более числовое значение (количество параметров заранее не определено). На основе этих значений рассчитываются накопленные суммы, которые сохраняются в списке, список возвращается как результат функции. Необязательный булевский параметр `mul` должен позволять заменять суммирование умножением.

2 семестр

1. Создать иерархию классов для фруктов, продающихся в магазине. Иерархия должна содержать не менее 3 классов. Объекты должны содержать не менее 2-х атрибутов и 2-х методов. Реализовать механизм автоматического подсчета количества всех созданных фруктов и автоматического присвоения каждому фрукту уникального идентификатора. Необходимо заполнить список представителями всех классов (всего не менее 10 объектов) и продемонстрировать работу созданного механизма.

2. Создайте класс `Length` (Длина), имеющий свойства: • `value` (значение), • `unit` (единица измерения). При изменении единицы измерения значение должно соответственно меняться. Например, при переходе от сантиметров к метрам значение должно уменьшаться в 100 раз. Допустимые значения свойства `unit`: 'см', 'м', 'км'. Организуйте эту проверку. Продемонстрируйте работу с классом.

3. Создайте класс `Заказ(Order)`, у которого есть свойства `код_товара(code)`, `цена(price)`, `количество(count)` и методы `__init__` и `__str__`. Создайте 2 класса-потомка: `Опт(Opt)` и `Розница(Retail)`. В этих классах создайте методы `__init__`, `__str__` и `сумма_заказа(summa)`,

позволяющий узнать стоимость заказа. Для опта стоимость единицы товара составляет 95% от цены, а при покупке более 500 штук – 90% цены. В розницу стоимость единицы товара составляет 100% цены. Стоимость заказа равна произведению цены на количество. Продемонстрируйте работу с классами, создав необходимые объекты и обратившись к их свойствам и методам.

4. Написать функцию-генератор `my_func_2(lst)`, которая принимает объект, поддерживающий итерации с произвольным уровнем вложенности, и возвращает все элементы по одному.

5. С помощью механизма `map/filter/reduce` (хотя бы одна из этих функций должна быть использована в решении) посчитать в тексте количество слов, состоящих не менее, чем из 3-х букв. Слова в тексте разделены пробелами. Написать реализацию в одну строку. Оформить решение в виде функции `my_func_3(text)`, т.е. шаблон таков: строка с `import`, если необходимо `def my_func_3(text): return #` однострочная реализация задания.

6. Написать декоратор с параметром `my_decorator(n)`. Декоратор превращает функцию, возвращающую поддерживающий итерации объект (далее "последовательность"), в функцию-генератор. Если декорируемая функция возвращает что-то другое, а не последовательность, то декоратор должен вернуть этот результат вызова функции без изменений. Проверку объекта можно организовать при помощи условия `import collections if isinstance (item, collections.Iterable)`. Параметром декоратора может быть целое положительное число `n`, тогда получившаяся функция-декоратор должна генерировать по одному значению из последовательности, повторенной `n` раз. Также параметр может принимать строковое значение `'inf'`, тогда функция-декоратор генерирует по одному значению из последовательности, повторенной бесконечное число раз (зацикливает генерирование результата). Подсказка: сначала реализовать случай со значением аргумента `'inf'`, а затем модифицировать для целочисленного значения аргумента.

Примерные вопросы для подготовки к Экзамену, Экзамену

Примерные вопросы для подготовки к экзамену (1 семестр)

Тема 1. Введение в программирование на Python

1. Присвоение по ссылке и по значению. Специфика создания объектов и присвоения в Python , особенности Python в связи с распространенностью использования неизменяемых типов.
2. Специфика типизации в языках программирования (различные аспекты типизации). Реализация типизации в Python .

Тема 2. Управляющие конструкции, списки и кортежи

3. Циклы в Python, работа и устройство цикла for, типичное применение range и enumerate в цикле for.
4. Списки в Python. Обращение к элементам списка и создание срезов. Обход списка и поиск элементов в списке. Ключевые операции, проводящие к изменению списка и порождающие измененные списки, копирование списков.

Тема 3. Словари, множества и выражения-генераторы

5. Словари в Python . Итерирование по словарям, преобразование между словарями и списками в Python. Операции с представлениями словарей.
6. Операции со словарями, учитывающие возможное отсутствие ключа. Операции многоэлементного изменения словарей. Операции поэлементного извлечения из словаря и их использование.
7. Множества в Python. Основные способы создания, получения и изменения значений. Обход множеств. Выполнение основных операций с парой множеств в Python.
8. Кортежи в Python. Отличия кортежей от списков. Распаковка и частичная распаковка кортежей.
9. Выражения генераторы и генераторы списков в Python. Использование условий в генераторах.
10. Функции стандартной библиотеки для работы с контейнерами.

Тема 4. Функции

11. Объявление и вызов функции в Python. Параметры функции со значением по умолчанию и комментирование функции. Получение информации о функции. Способы передачи параметров при вызове функции.

12. Передача переменного количества параметров (именованных и не именованных) в функции Python. Вызов функции с позиционными параметрами, находящимися в списке, и именованными параметрами, находящимися в словаре.

Тема 5. Работа с файлами и обработка исключительных ситуаций

13. Синтаксис и семантика обработки исключительных ситуаций в Python. Создание пользовательских исключений и инструкция `assert`.

14. Базовые операции для работы с файлами в Python. Использование инструкции `with ... as` на примере работы с файлами.

Тема 6. Модули и пакеты

15. Модули в Python и их отличие от скриптов Python. Варианты синтаксиса импорта модуля и объектов модуля. Применение импортированных объектов. Порядок поиска модулей и специфика их загрузки. Загрузка модулей из глобального репозитория.

Пример экзаменационного билета (1 семестр)

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ №

1. Присвоение по ссылке и по значению. Специфика создания объектов и присвоения в Python, особенности Python в связи с распространенностью использования неизменяемых типов. **(20 баллов)**

2. На основе строки, представляющей из себя предложение, построить вложенный список, содержащий символы всех слов в предложении. Пример: строка 'Eeny, meeny, miney, moe; Catch a tiger by his toe.' будет преобразована в: `[['E', 'e', 'n', 'y'], ['m', 'e', 'e', 'n', 'y'], ['m', 'i', 'n', 'e', 'y'], ['m', 'o', 'e'], ['C', 'a', 't', 'c', 'h'], ['a'], ['t', 'i', 'g', 'e', 'r'], ['b', 'y'], ['h', 'i', 's'], ['t', 'o', 'e']]` **(20 баллов)**

3. Создать иерархию классов для фруктов, продающихся в магазине. Иерархия должна содержать не менее 3 классов. Объекты должны содержать не менее 4х атрибутов. Часть атрибутов должна быть защищена от изменения, а часть и от изменения, и от чтения. Необходимо заполнить список представителями всех классов (всего не менее 10 объектов) и продемонстрировать созданную защиту. **(20 баллов)**

Примерные вопросы для подготовки к экзамену (2 семестр)

Тема 9-10. Объектно-ориентированное программирование

1. Концепция класса и объекта. Принципы и механизмы ООП.
2. Объявление класса, конструктор, создание объектов и одиночное наследование в Python. Управление доступом к атрибутам класса в Python.
3. Полиморфизм и утиная типизация и проверка принадлежности объекта к классу в языке Python.
4. Методы классов и статические переменные и методы в Python. Специальные методы для использования пользовательских классов со стандартными операторами и функциями.

Тема 11-12. Функциональное программирование

5. Основные возможности, поддерживаемые функциональными языками программирования. Поддержка элементов функционального программирования в Python.
6. Концепция «функции – граждане первого класса» в языке программирования, поддержка этой концепции в Python. Специфика лямбда-функций в Python их возможности и ограничения. Типичные сценарии использования лямбда-функций в Python.
7. Глобальные и локальные переменные в функциях на примере Python. Побочные эффекты вызова функций и их последствия.
8. Вложенные функции и замыкания, специфика реализации в Python.
9. Функции высшего порядка и декораторы в Python.

10. Концепция map/filter/reduce. Реализация map/filter/reduce в Python и пример их использования.

11. Итераторы в Python: встроенные итераторы, создание собственных итераторов, типичные способы обхода итераторов и принцип их работы. Встроенные функции для работы с итераторами и возможности модуля itertools.

12. Функции генераторы и выражения генераторы: создание и применение в Python.

Тема 13. Структуры данных: массивы, стеки, очереди, списки.

13. Специфика массивов, как структур данных. Динамические массивы – специфика работы, сложность операций. Специфика работа с array в Python.

14. Абстрактная структура данных стек и очередь: базовые и расширенные операции, их сложность.

15. Специфика реализации и скорости основных операций в очереди на базе массива и связанного списка.

16. Связанные списки: однонаправленные и двунаправленные – принцип реализации. Сравнение скорости выполнения основных операций в связанных списках и в динамическом массиве.

Тема 14. Алгоритмы поиска и сортировки

17. Алгоритм обменной сортировки, сложность сортировки и возможности по ее улучшению.

18. Алгоритм сортировки выбором, сложность сортировки и возможности по ее улучшению.

19. Алгоритм сортировки вставками, его сложность. Алгоритм быстрого поиска в отсортированном массиве. Сложность поиска в отсортированном и не отсортированном массиве.

20. Алгоритм сортировки Шелла, сложность сортировки и возможности по ее улучшению.

21. Алгоритм быстрой сортировки, сложность сортировки и возможности по ее улучшению.

22. Алгоритм сортировки слиянием, сложность сортировки.

Тема 15. Структуры данных: деревья

23. Реализация двоичных деревьев в виде связанных объектов. Различные реализации рекурсивного обхода двоичных деревьев.

24. Двоичное дерево поиска – принципы реализации и логика реализации основных операций.

25. Двоичная куча – принципы реализации и логика реализации основных операций.

Тема 16. Хеш-таблицы

26. Абстрактный тип данных - ассоциативный массив и принцип его реализации на основе хэш-таблиц и хэш-функций.

27. Общая схема построения хэш-функции и возможная роль в этой схеме хэш-функции multiply-add-and-divide. Принцип работы хэш-функции multiply-add-and-divide.

28. Полиномиальная хэш-функция – принцип работы, специфика эффективной реализации и специфика применения хэш-функции.

29. Различные методы разрешения коллизий в хэш-таблицах.

Пример экзаменационного билета (2 семестр)

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ №

1. Специфика типизации в языках программирования (различные аспекты типизации). Реализация типизации в Python . **(20 баллов)**

2. В строке содержащей последовательность слов, разделенных запятыми удалить все нечетные слова. Ответ представить в виде строки. Пример: строка 'SIX,SEVEN,EIGHT,NINE,TEN' будет преобразована в: 'SIX,EIGHT,TEN'. **(20 баллов)**

3. Создать иерархию классов для фруктов, продающихся в магазине. Иерархия должна содержать не менее 3 классов. Объекты должны содержать не менее 2х атрибутов и 2х методов. Реализовать механизм автоматического подсчета количества всех созданных фруктов и автоматического присвоения каждому фрукту уникального идентификатора. Необходимо заполнить список

представителями всех классов (всего не менее 10 объектов) и продемонстрировать работу созданного механизма. **(20 баллов)**

8. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины

Нормативно-правовые акты:

- не предусмотрены

Основная литература:

1. Федоров , Дмитрий Юрьевич Программирование на python : учебное пособие для вузов / Д. Ю. Федоров. 6-е изд. , пер. и доп Электрон. дан. Москва : Юрайт , 2025 187 с (Высшее образование) URL: <https://urait.ru/bcode/556864> (дата обращения: 08.04.2025). Режим доступа: Электронно-библиотечная система Юрайт, для авториз. пользователей <https://urait.ru/bcode/556864> ISBN 978-5-534-19666-5 : 829.00
2. Гуриков , Сергей Ростиславович Основы алгоритмизации и программирования на Python : Учебное пособие / Московский технический университет связи и информатики 1 Москва : ООО "Научно-издательский центр ИНФРА-М" , 2025 343 с. (Высшее образование) ВО - Бакалавриат <https://znanium.ru/catalog/document?id=453296> ISBN 978-5-16-020255-6 ISBN 978-5-16-102278-8 (электр. издание)

3. Практикум по программированию на языке Python : учебное пособие для студентов, обучающихся по направлениям "Прикладная математика и информатика", "Прикладная информатика", "Информационная безопасность", "Бизнес-информатика" (программа подготовки бакалавра) / Р.И. Горохова, Е.П. Догадина, В.И. Долгов, В.И. Макрушин ; Финуниверситет, Департамент анализа данных и машинного обучения факультета информационных технологий и анализа больших данных Москва : Финуниверситет , 2023 1 файл (6,93 Мб) Свободный доступ из сети Интернет (чтение, печать, копирование) (дата обращения 09.10.2023) + Текст : электронный

Дополнительная литература:

1. Программирование. Основы Python для инженеров [Электронный ресурс] : учебное пособие для вузов / Никитина Т. П., Королев Л. В. ; Королев Л. В. 2-е изд., стер. Санкт-Петербург : Лань , 2025 156 с. Книга из коллекции Лань - Информатика <https://e.lanbook.com/book/454463> ISBN 978-5-507-50668-2

2. Алгоритмы и структуры данных на Python : Учебное пособие / С.А. Чернышев Электрон. дан. Москва : КноРус , 2024 326 с. Режим доступа: book.ru Internet access <https://book.ru/book/949701> ISBN 978-5-406-11683-8

3. Языки программирования. Python: решение сложных задач [Электронный ресурс] : учебное пособие для вузов / Борзунов С. В., Кургалин С. Д. Санкт-Петербург : Лань , 2023 192 с. Книга из коллекции Лань - Информатика <https://e.lanbook.com/book/319394> ISBN 978-5-507-45923-0

4. Чернышев , Станислав Андреевич Основы программирования на Python : учебник для вузов / С. А. Чернышев. 2-е изд. , пер. и доп Электрон. дан. Москва : Юрайт , 2025 349 с (Высшее образование) URL: <https://urait.ru/bcode/567821> (дата обращения: 08.04.2025). Режим доступа: Электронно-библиотечная система Юрайт, для авториз. пользователей <https://urait.ru/bcode/567821> ISBN 978-5-534-17139-6 : 1729.00

5. Шелудько , Виктория Михайловна Основы программирования на языке высокого уровня Python : Учебное пособие Ростов-на-Дону : Издательство Южного федерального университета (ЮФУ) , 2017 146

с. ВО - Бакалавриат <https://znanium.com/catalog/document?id=339834> ISBN 978-5-9275-2649-9

9. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

1. Электронная библиотека Финансового университета (ЭБ) <http://elib.fa.ru/> (<http://library.fa.ru/files/elibfa.pdf>)
2. Python Documentation <http://python.org/doc/>
3. Python Standard Library <https://docs.python.org/2/library/>
4. Информационно-образовательный портал Финуниверситета: <https://org.fa.ru>
5. Электронно-библиотечная система "Университетская библиотека ОНЛАЙН" <http://biblioclub.ru/>
6. Электронный учебный курс <https://campus.fa.ru/course/view.php?id=12139>
7. Личный кабинет обучающегося <https://org.fa.ru>
8. Федеральная ЭБС «Единое окно доступа к образовательным ресурсам». – URL: <http://window.edu.ru>
9. Интернет-репозиторий образовательных ресурсов – URL: <http://repository.vzfei.ru>
10. Базовый ЭУК по дисциплине на сайте campus.fa.ru.
11. <http://pandas.pydata.org/>
12. Библиотечно-информационный комплекс Финуниверситета (электронная библиотека, ресурсы на русском языке): http://www.library.fa.ru/res_mainres.asp?cat=rus

10. Методические указания для обучающихся по освоению дисциплины

Студентам при подготовке следует использовать нормативные документы Финансового университета, Методические рекомендации по планированию и организации внеаудиторной самостоятельной работы студентов по образовательным программам бакалавриата и магистратуры в Финансовом университете, утвержденные приказом Финуниверситета от 11.05.2021 г. № 1040/о (см. сайт Финансового Университета: на главной странице раздел "Наш университет" → "Единая правовая база Финуниверситета" → "Организация учебного процесса" → "Нормативные документы по самостоятельной работе").

Лекционные занятия проводятся в соответствии с тематическим планом, при изложении материала рекомендуется использовать презентации в среде PowerPoint, программный код из Jupyter Notebook и материалы по теме лекции.

В ходе интерактивных занятий следует проводить разбор конкретных примеров программного кода из Jupyter Notebook.

11. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень необходимого программного обеспечения и информационных справочных систем

- Комплект лицензионного программного обеспечения:

1. Python 3.8
2. Пакет офисных программ
3. Kaspersky

- Современные профессиональные базы данных и информационные справочные системы:

1. Информационно-правовая система «Гарант»
2. Информационно-правовая система «Консультант Плюс»

3. Электронная энциклопедия: <http://ru.wikipedia.org/wiki/Wiki>

4. Система комплексного раскрытия информации «СКРИН» - <http://www.skrin.ru/>

- Сертифицированные программные и аппаратные средства защиты информации

1. не предусмотрены

12. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине

1. **Компьютерный класс** для проведения учебных занятий, предусмотренных программой, в том числе групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, оснащенный оборудованием и техническими средствами обучения: мебель аудиторная (столы, стулья, доска аудиторная), персональные компьютеры, набор демонстрационного оборудования (проектор, экран)

2. **Учебная аудитория** для проведения учебных занятий, предусмотренных программой, в том числе групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, оснащенная оборудованием и техническими средствами обучения: мебель аудиторная (столы, стулья, доска аудиторная), набор демонстрационного оборудования (проектор, экран)

3. **Помещение для самостоятельной работы** обучающихся оснащенное компьютерной техникой с возможностью подключения к сети "Интернет" и обеспечением доступа к электронной информационно-образовательной среде Университета.