

**Федеральное государственное образовательное бюджетное  
учреждение высшего образования  
«ФИНАНСОВЫЙ УНИВЕРСИТЕТ  
ПРИ ПРАВИТЕЛЬСТВЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»  
(Финансовый университет)**

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

УТВЕРЖДАЮ

Проректор по учебной и  
методической работе

\_\_\_\_\_ Е.А. Каменева

29.11.2024 г.

**Горохова Р.И., Догадина Е.П., Долгов В.И.**

**Алгоритмы и структуры данных в языке Python**

**Рабочая программа дисциплины**

для студентов, обучающихся по направлению подготовки:  
09.03.03 - Прикладная информатика,  
ОП «Прикладные информационные системы в экономике и финансах»

*Рекомендовано Ученым советом  
Факультета информационных технологий и анализа больших данных  
(протокол № 49 от 19.11.2024 г.)*

*Одобрено советом Кафедры информационных технологий  
(протокол № 4 от 12.11.2024 г.)*

**Москва 2024**

## Содержание

1. Наименование дисциплины.....	3
2. Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине .....	3
3. Место дисциплины в структуре образовательной программы .....	4
4. Объем дисциплины (модуля) в зачетных единицах и в академических часах с выделением объема аудиторной (лекции, семинары) и самостоятельной работы обучающихся .....	5
5. Содержание дисциплины, структурированное по темам (разделам) дисциплины с указанием их объемов (в академических часах) и видов учебных занятий.....	5
5.1. Содержание дисциплины.....	5
5.2. Учебно-тематический план.....	10
5.3. Содержание семинаров, практических занятий.....	12
6. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине.....	15
6.1. Перечень вопросов, отводимых на самостоятельное освоение дисциплины, формы внеаудиторной самостоятельной работы.....	15
6.2. Перечень вопросов, заданий, тем для подготовки к текущему контролю...	17
7. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине.....	23
8. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины.....	33
9. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины.....	34
10. Методические указания для обучающихся по освоению дисциплины.....	36
11. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень необходимого программного обеспечения и информационных справочных систем.....	36
12. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине.....	37

## 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	Индикаторы достижения компетенции	Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции
ПКН-2	Способность разрабатывать алгоритмы и программы с использованием современных технологий программирования.	1. Владеет объектно-ориентированным языком программирования на уровне знания синтаксиса и семантики, основ стандартной библиотеки.	<b>Знать:</b> объектно-ориентированный язык программирования Python на уровне знания синтаксиса и семантики, основ стандартной библиотеки. <b>Уметь:</b> определять на уровне знания синтаксиса и семантику, стандартные библиотеки языка Python, необходимые для решения прикладных задач.
		2. Использует инструментальные средства программирования (IDE, SDK, API, популярные фреймворки и библиотеки).	<b>Знать:</b> инструментальные средства программирования (IDE, SDK, API, популярные фреймворки и библиотеки). <b>Уметь:</b> разрабатывать программы решения задач с использованием инструментальных средств программирования (IDE, SDK, API, популярных фреймворков и библиотек).
		3. Организует кодовую базу, ориентируется в существующем коде, демонстрирует знание общепринятых соглашений и политик в области оформления кода.	<b>Знать:</b> особенности создания программного кода. <b>Уметь:</b> разрабатывать программный код, ориентироваться в существующем коде, применять знание общепринятых соглашений и политик в области оформления кода.
		4. Проектирует текстовый, программный или графический интерфейс	<b>Знать:</b>

		программной системы исходя из ее назначения.	основы проектирования различные виды интерфейса программной системы. <b>Уметь:</b> разрабатывать текстовый, программный или графический интерфейс программной системы исходя из ее назначения.
ПКН-3	Способность проектировать и реализовывать архитектуру и дизайн программной системы в соответствии с анализом задачи и требований к ней.	1. Демонстрирует знание основных алгоритмов и структур данных, использует на практике простые структуры данных, оценивает сложность алгоритмов.	<b>Знать:</b> основные алгоритмы и базовые структуры данных.  <b>Уметь:</b> разрабатывать алгоритмы для работы со структурами данных, оценивать сложность алгоритмов.
		2. Собирает, формулирует, систематизирует и анализирует функциональные и нефункциональные требования к информационной системе, выбирает архитектурные решения на их основе.	<b>Знать:</b> требования к информационной системе, архитектурные решения на их основе.  <b>Уметь:</b> выбирать архитектурные решения разработки информационных систем на основе систематизации и анализа функциональных и нефункциональных требований к ним.
		3. Создает объектно-ориентированный код, инкапсулирующий условия задачи, производит декомпозицию задачи и проектирует систему в пределах одной платформы или технологии.	<b>Знать:</b> принципы разработки объектно-ориентированного кода. <b>Уметь:</b> создавать объектно-ориентированный код, производить декомпозицию задачи и проектировать систему в пределах одной платформы или технологии.

### 3. Место дисциплины в структуре образовательной программы

Дисциплина «Алгоритмы и структуры данных в языке Python» относится к Общепрофессиональному циклу дисциплин по направлению подготовки 09.03.03 - Прикладная информатика, ОП "Прикладные информационные системы в экономике и финансах".

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

заочная форма обучения (ИОО)

Вид учебной работы по дисциплине	Всего (в з/е и часах)	Семестр 2 (в часах)	Семестр 3 (в часах)
Общая трудоемкость дисциплины	8/288	144	144
Контактная работа – Аудиторные занятия	32	16	16
Лекции	6	4	2
Семинары, практические занятия	26	12	14
Самостоятельная работа	256	128	128
Вид текущего контроля		контрольная работа	контрольная работа
Вид промежуточной аттестации		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. Именованные переменные.

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 упаковка 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: создание 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

№ п/п	Наименование тем (разделов) дисциплины	Трудоемкость в часах					Формы текущего контроля успеваемости
		Все го	*Контактная работа - Аудиторная работа			Само- стоя- тельная работа	
			Об- щая, в т.ч.:	Лек- ции	Семинары, практиче- ские заня- тия		
1.	Введение в про- граммирование на Python	19	3	2	1	16	Самостоятельное реше- ние задач (программиро- вание), выступление на семинаре
2.	Управляющие конструкции, списки и кор- тежи	17	1	-	1	16	Самостоятельное реше- ние задач (программиро- вание), выступление на семинаре
3.	Словари, множе- ства и выраже- ния-генераторы	18	2	-	2	16	Самостоятельное реше- ние задач (программиро- вание), выступление на семинаре
4.	Функции	18	2	-	2	16	Самостоятельное реше- ние задач (программиро- вание), выступление на семинаре
5.	Работа с фай- лами и обра- ботка исключи- тельных ситуа- ций	20	4	2	2	16	Самостоятельное реше- ние задач (программиро- вание), выступление на семинаре
6.	Модули и па- кеты	18	2	-	2	16	Самостоятельное реше- ние задач (программиро- вание), выступление на семинаре
7.	Продвинутые коллекции	18	2	-	2	16	Самостоятельное реше- ние задач (программиро- вание), выступление на семинаре

8.	Обзор современных языков программирования	16	-	-	-	16	Самостоятельное решение задач (программирование), выступление на семинаре
9.	Введение в объектно-ориентированное программирование	20	4	2	2	16	Самостоятельное решение задач (программирование), выступление на семинаре
10.	Объектно-ориентированное программирование в Python	18	2	-	2	16	Самостоятельное решение задач (программирование), выступление на семинаре
11.	Введение в функциональное программирование	17	1	-	1	16	Самостоятельное решение задач (программирование), выступление на семинаре
12.	Функциональное программирование в Python	17	1	-	1	16	Самостоятельное решение задач (программирование), выступление на семинаре
13.	Структуры данных: массивы, стеки, очереди, списки	18	2	-	2	16	Самостоятельное решение задач (программирование), выступление на семинаре
14.	Алгоритмы поиска и сортировки	18	2	-	2	16	Самостоятельное решение задач (программирование), выступление на семинаре
15.	Структуры данных: деревья	18	2	-	2	16	Самостоятельное решение задач (программирование), выступление на семинаре
16.	Хеш-таблицы	18	2	-	2	16	Самостоятельное решение задач (программирование), выступление на семинаре
	В целом по дисциплине	288	32	6	26	256	Согласно учебному плану: контрольные работы
	Итого в %		11	19	81	89	

\* объем контактной работы в очно-заочной/заочной формах обучения и индивидуальных учебных планах определяется соответствующими учебными планами. Темы, реализуемые в виде контактной работы, определяются преподавателем самостоятельно, исходя из уровня их сложности.

### 5.3. Содержание семинаров, практических занятий

Наименование тем (разделов) дисциплины	Перечень вопросов для обсуждения на семинарских, практических занятиях, рекомендуемые источники из разделов 8,9 (указывается раздел и порядковый номер источника)	Формы проведения занятий
1. Введение в программирование на Python	Установка Python, установка дистрибутива Anaconda. Работа в интерактивном режиме интерпретатора. Интерактивная оболочка IPython notebook: принципы работы и применения.  Источники: 8.1; 8.2; 8.3	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
2. Управляющие конструкции, списки и кортежи	Базовые числовые типы, строки, списки, словари, переменные, базовые операторы.  Источники: 8.1; 8.2; 8.3	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
3. 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.  Источники: 8.1; 8.2; 8.3	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
4. Функции	Создание функций, область видимости переменной, передача аргументов в функцию. Лямбда-функции.  Источники: 8.1; 8.2; 8.3	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)

5. Работа с файлами и обработка исключительных ситуаций	<p>Исключения. Инструкция <code>try...except...else...finally</code>. Классы встроенных исключений. Создание пользовательских исключений. Инструкция <code>assert</code>.</p> <p>Работа с файлами в Python, операции с файлами: открытие/закрытие файла, чтение и записи и другие методы для работы с файлами. Инструкция <code>with ... as</code> и ее использование для файлов.</p> <p>Источники: 8.1; 8.2; 8.3</p>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
6. Модули и пакеты	<p>Устройство модулей и пакетов, инструкции <code>import</code> и <code>from</code>. Создание собственных модулей и пакетов.</p> <p>Источники: 8.1; 8.2; 8.3</p>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
7. Продвинутое коллекции	<p>Модули стандартной библиотеки Python, предоставляющие дополнительный функционал по работе с коллекциями объектов. Класс <code>Frozen set</code>. Модуль <code>collections</code></p> <p>Источники: 8.1; 8.2; 8.3</p>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
8. Обзор современных языков программирования	<p>Парадигмы программирования. Специализация языков программирования. Место Python в современном ландшафте языков программирования.</p> <p>Характеристики языков программирования: способ передачи параметров в функцию; способ управления динамической памятью.</p> <p>Источники: 8.1; 8.2; 8.3</p>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
9. Введение в объектно-ориентированное программирование	<p>Базовые возможности ООП в Python: создание классов и объектов; наследование и полиморфизм; функция <code>super()</code>.</p> <p>Источники: 8.1; 8.2; 8.3</p>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)

10. Объектно-ориентированное программирование в Python	<p>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.</p> <p>Источники: 8.1; 8.2; 8.3</p>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
11. Введение в функциональное программирование	<p>Элементы функционального программирования в Python: функции – граждане первого класса; глобальные и локальные переменные в Python; вложенные функции и замыкания в Python.</p> <p>Декораторы в Python: использование и создание собственных декораторов.</p> <p>Источники: 8.1; 8.2; 8.3</p>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
12. Функциональное программирование в Python	<p>Реализация функций map, filter, reduce в Python.</p> <p>Итераторы в Python, итерируемый тип данных. Модуль itertools.</p> <p>Источники: 8.1; 8.2; 8.3</p>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
13. Структуры данных: массивы, стеки, очереди, списки	<p>Массивы и их отличие от списков в Python. Динамические массивы, сложность операций работы с динамическими массивами.</p> <p>Стек, операции со стеком. Реализации стека. Очередь, операции с очередью. Реализация очереди. Связные списки, варианты связанных списков.</p> <p>Источники: 8.1; 8.2; 8.3</p>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
14. Алгоритмы поиска и сортировки	<p>Реализация на Python простых методов сортировки: обменные сортировки (с различными вариациями); сортировка выбором (извлечением); сортировка включением (вставками).</p> <p>Реализация на Python эффективных методов сортировки: быстрая сортировка; сортировка Шелла; сортировка слиянием.</p> <p>Источники: 8.1; 8.2; 8.3</p>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)

15. Структуры данных: деревья	Реализация на Python деревьев, бинарных деревьев. Использование бинарных деревьев в прикладных задачах: представление выражения (предложения) в виде дерева.  Источники: 8.1; 8.2; 8.3	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (не менее 30% времени на интерактивные технологии)
16. Хеш-таблицы	Работа с функцией hash в Python. Использование специализированных библиотек для работы с хэш-функциями.  Источники: 8.1; 8.2; 8.3	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
1. Введение в программирование на Python	Среда программирования. Использование документации.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
2. Управляющие конструкции, списки и кортежи	Оперирование вложенными списками, копирование списков, некоторые операции над списками.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
3. Словари, множества и выражения-генераторы	Выражения-генераторы в Python. Выражения-генераторы для словарей и множеств: семантика, синтаксис и практическое использование.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
4. Функции	Возвращение нескольких значений из функции. Распаковка и упаковка параметров функции.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.

5. Работа с файлами и обработка исключительных ситуаций	Сохранение объектов в файл с помощью модуля pickle и shelve. Модуль CSV.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
6. Модули и пакеты	Написание и запуск скриптов на Python. Установка модулей из глобального репозитория.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
7. Продвинутое коллекция	Класс Counter, класс defaultdict, класс OrderedDict, класс namedtuple. Модуль enum.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
8. Обзор современных языков программирования	Виды типизации переменных в различных современных языках программирования.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
9. Введение в объектно-ориентированное программирование	Базовые типы в Python: взгляд с точки зрения ООП. Методы базовых типов.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
10. Объектно-ориентированное программирование в Python	Проверка принадлежности к классу и интроспекция в Python.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
11. Введение в функциональное программирование	Реализация декораторов с параметрами в Python.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
12. Функциональное программирование в Python	Функции-генераторы и выражения-генераторы в Python.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
13. Структуры данных: массивы, стеки, очереди, списки	Реализация различных вариантов связанных списков на Python.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
14. Алгоритмы поиска и сортировки	Сравнение различных сортировок (простых и эффективных) с использованием их реализаций на Python.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
15. Структуры данных: деревья	Реализация обхода бинарных деревьев.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.
16. Хеш-таблицы	Использование специализированных библиотек для работы с хэш-функциями: проверка хэш-конфликтов для различных наборов данных.	Индивидуальное выполнение заданий с использованием Jupyter Notebook.



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

### ***Примерные вопросы к контрольной работе 2 семестра***

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. Упаковка 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 задания контрольной работы 2 семестра***

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'}

4. 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'}.
5. 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).

6. Реализовать функцию `summlate` 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.

### ***Примерные вопросы к контрольной работе 3 семестра***

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.

### ***Примерные задания контрольной работы 3 семестра***

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 (за циклирует генерирование результата).  
Подсказка: сначала реализовать случай со значением аргумента 'inf', а затем модифицировать для целочисленного значения аргумента.

*Критерии балльной оценки различных форм текущего контроля успеваемости содержатся в соответствующих методических рекомендациях Кафедры информационных технологий Факультета информационных технологий и анализа больших данных.*

## **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**

<b>Наименование компетенции</b>	<b>Наименование индикаторов достижения компетенции</b>	<b>Результаты обучения (умения и знания), соотнесенные с индикаторами достижения компетенции</b>	<b>Типовые контрольные задания</b>
Способность разрабатывать алгоритмы и программы с использованием современных технологий программирования (ПКН-2)	1. Владеет объектно-ориентированным языком программирования на уровне знания синтаксиса и семантики, основ стандартной библиотеки.	<b>Знать:</b> объектно-ориентированный язык программирования Python на уровне знания синтаксиса и семантики, основ стандартной библиотеки.	Напишите программу на Python, обрабатывающую текстовую строку так, чтобы в ней все заглавные буквы преобразовать в строчные.

		<p><b>Уметь:</b> определять на уровне знания синтаксис и семантику, стандартные библиотеки языка Python, необходимые для решения прикладных задач.</p>	<p>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.</p>
	<p>2.Использует инструментальные средства программирования (IDE, SDK, API, популярные фреймворки и библиотеки).</p>	<p><b>Знать:</b> инструментальные средства программирования (IDE, SDK, API, популярные фреймворки и библиотеки).</p> <p><b>Уметь:</b> разрабатывать программы решения задач с использованием инструментальных средств программирования (IDE, SDK, API, популярных фреймворков и библиотек).</p>	<p>Используя программу Jupyter Notebook составьте код на Python, определяющий количество точек на плоскости, находящихся на заданном расстоянии от начала координат. Используйте библиотеку math.</p> <p>Выберите библиотеку Python для работы с массивами. Выполните программный код бинарного поиска данных.</p>
	<p>3.Организовывает кодовую базу, ориентируется в существующем коде, демонстрирует знание общепринятых соглашений и политик в области оформления кода.</p>	<p><b>Знать:</b> особенности создания программного кода.</p> <p><b>Уметь:</b> разрабатывать программный код, ориентироваться в существующем коде, применять знание общепринятых соглашений и политик в области оформления кода.</p>	<p>Выберите библиотеку Python для создания и обработки деревьев данных.</p> <p>Выполните программный код вычисления арифметического выражения с помощью двоичного дерева.</p>
	<p>4.Проектирует текстовый, программный или графический интерфейс программной системы исходя из ее назначения.</p>	<p><b>Знать:</b> основы проектирования различных видов интерфейса программной системы.</p>	<p>Реализуйте программный код вычисления различных арифметических операций, выбор которых осуществляется с помощью разработанного меню.</p>



		<b>Уметь:</b> разрабатывать текстовый, программный или графический интерфейс программной системы исходя из ее назначения.	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.
Способность проектировать и реализовывать архитектуру и дизайн программной системы в соответствии с анализом задачи и требований к ней (ПКН-3)	1. Демонстрирует знание основных алгоритмов и структур данных, использует на практике простые структуры данных, оценивает сложность алгоритмов.	<b>Знать:</b> основные алгоритмы и базовые структуры данных.  <b>Уметь:</b> разрабатывать алгоритмы для работы со структурами данных, оценивать сложность алгоритмов.	Выполните обработку текстового документа, 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.  Опишите структуру CSV файла фиксирующего факты продаж, необходимого для подсчета суммарных продаж по различным географическим подразделениям фирмы.
	2. Собирает, формулирует, систематизирует и анализирует функциональные и нефункциональные требования к информационной системе, выбирает архитектурные решения на их основе.	<b>Знать:</b> требования к информационной системе, архитектурные решения на их основе.  <b>Уметь:</b> выбирать архитектурные решения разработки информационных систем на основе систематизации и анализа функциональных и нефункциональных требований к ним.	Определите закономерности в синтаксических конструкциях Python для визуализации данных с применением графической библиотеки.  Написать скрипт, который заменяет в текстовом файле все слова из определенного перечня на определенные для этих слов слова-заменители. Подсчитать количество замен и количество символов в заменах.
	3. Создает объектно-ориентиро-	<b>Знать:</b>	Создайте класс Length (Длина), имеющий свойства:

	<p>ванный код, инкапсулирующий условия задачи, производит декомпозицию задачи и проектирует систему в пределах одной платформы или технологии.</p>	<p>принципы разработки объектно-ориентированного кода.</p> <p><b>Уметь:</b> создавать объектно-ориентированный код, производить декомпозицию задачи и проектировать систему в пределах одной платформы или технологии.</p>	<ul style="list-style-type: none"> <li>• value (значение),</li> <li>• unit (единица измерения). Разработайте класс наследник, для вычисления площади различных фигур.</li> </ul> <p>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.</p>
--	--	--	---

### *Примерные вопросы для подготовки к экзамену (2 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. Организация ветвлений в Python. Инструкция if...else. Множественный выбор.
4. Циклы в Python, работа и устройство цикла for, типичное применение range и enumerate в цикле for.
5. Циклы в Python, работа и устройство цикла while. Организация бесконечного цикла, аварийное завершение цикла.
6. Списки в Python. Обращение к элементам списка и создание срезов. Обход списка и поиск элементов в списке.

7. Ключевые операции, проводящие к изменению списка и порождающие измененные списки, копирование списков.

### **Тема 3. Словари, множества и выражения-генераторы**

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

### **Тема 4. Функции**

14. Объявление и вызов функции в Python. Параметры функции со значением по умолчанию и комментирование функции.
15. Функция. Получение информации о функции. Способы передачи параметров при вызове функции.
16. Передача переменного количества параметров (именованных и не именованных) в функции Python.
17. Вызов функции с позиционными параметрами, находящимися в списке, и именованными параметрами, находящимися в словаре.
18. Вложенные функции и замыкания, специфика реализации в Python.
19. Функции высшего порядка и декораторы в Python.
20. Концепция map/filter/reduce. Реализация map/filter/reduce в Python и пример их использования.

21. 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`.
22. Функции генераторы и выражения генераторы: создание и применение в Python.

### **Тема 5. Работа с файлами и обработка исключительных ситуаций**

23. Синтаксис и семантика обработки исключительных ситуаций в Python. Создание пользовательских исключений и инструкция `assert`.
24. Базовые операции для работы с файлами в Python. Использование инструкции `with ... as` на примере работы с файлами.
25. CSV файлы. Методы и функции работы с CSV файлами.
26. Бинарные файлы. Методы и функции работы с бинарными файлами.

### **Тема 6. Модули и пакеты**

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

### ***Пример экзаменационного билета***

**Федеральное государственное образовательное бюджетное учреждение  
высшего образования  
«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ  
РОССИЙСКОЙ ФЕДЕРАЦИИ»  
(Финансовый университет)**

Кафедра информационных технологий

Дисциплина **Алгоритмы и структуры данных в языке Python**

Факультет информационных технологий и анализа больших данных

Форма обучения: заочная

Семестр: 2

Направление подготовки: **09.03.03 - Прикладная информатика**

Профиль: Прикладные информационные системы в экономике и финансах

### **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, мое; 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)

Подготовил:

/ Ф.И.О./

Утверждаю:

Первый заместитель

руководителя департамента

\_\_\_\_\_/ Ф.И.О./

\_\_\_\_\_ Дата

### ***Примерные вопросы для подготовки к экзамену (3 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

Федеральное государственное образовательное бюджетное учреждение  
высшего образования  
«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ  
РОССИЙСКОЙ ФЕДЕРАЦИИ»  
(Финансовый университет)

Кафедра информационных технологий

Дисциплина **Алгоритмы и структуры данных в языке Python**

Факультет информационных технологий и анализа больших данных

Форма обучения: заочная

Семестр: 3

Направление подготовки: **09.03.03 - Прикладная информатика**

Профиль: Прикладные информационные системы в экономике и финансах

### 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)**

Подготовил:

/ Ф.И.О./

Утверждаю:

Первый заместитель

руководителя департамента

\_\_\_\_\_/ Ф.И.О./

\_\_\_\_\_ Дата



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

### **Основная литература:**

1. 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. — ЭБ Финуниверситета. — URL: <http://elib.fa.ru/fbook/books137296.pdf> (дата обращения: 25.10.2024). — Текст : электронный.

2. Гуриков, С. Р. Основы алгоритмизации и программирования на Python : учебное пособие / С.Р. Гуриков. — Москва : ИНФРА-М, 2025. — 343 с. — (Высшее образование). — ЭБС ZNANIUM - URL: <https://znanium.ru/catalog/product/2166199> (дата обращения: 25.10.2024). — Текст : электронный.

### **Дополнительная литература:**

3. Шелудько, В. М. Язык программирования высокого уровня Python. Функции, структуры данных, дополнительные модули : учебное пособие / В. М. Шелудько; Южный федеральный университет. - Ростов-на-Дону ; Таганрог : Южный федеральный университет, 2017. - 107 с. — ЭБС ZNANIUM. - URL: <https://znanium.com/catalog/product/1021664> (дата обращения: 25.10.2024). — Текст : электронный.

4. Северенс, Ч. Введение в программирование на Python / Ч. Северенс. — 2-е изд., испр. — Москва : Национальный Открытый Университет «ИНТУИТ», 2016. — 231 с. — ЭБС Университетская библиотека ONLINE. — URL: <https://biblioclub.ru/index.php?page=book&id=429184> (дата обращения: 25.10.2024). — Текст : электронный.

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

1. Pylru 1.0.9 [Электронный ресурс]: сайт. – Режим доступа: <https://pypi.python.org/pypi/pylru>.
2. Python Data Analysis Library [Электронный ресурс]: сайт. – Режим доступа: <http://pandas.pydata.org/>.
3. Python Documentation [Электронный ресурс]: сайт. – Режим доступа: <http://python.org/doc/>.
4. Python Standard Library [Электронный ресурс]: сайт. – Режим доступа: <https://docs.python.org/2/library/>.
5. Scikit-learn Machine Learning in Python [Электронный ресурс]: сайт. – Режим доступа: <http://scikit-learn.org>.
6. Официальный сайт продукта <https://www.python.org/>.
7. Портал Финансового университета <http://www.fa.ru/>.
8. Электронная библиотека Финансового университета (ЭБ) <http://elib.fa.ru/>
9. Электронно-библиотечная система BOOK.RU <http://www.book.ru>
10. Электронно-библиотечная система «Университетская библиотека ОНЛАЙН» <http://biblioclub.ru/>
11. Электронно-библиотечная система Znanium <http://www.znaniy.com>
12. Электронно-библиотечная система издательства «ЮРАЙТ» <https://urait.ru/>
13. Электронно-библиотечная система издательства Проспект <http://ebs.prospekt.org/books>
14. Электронно-библиотечная система издательства Лань <https://e.lanbook.com/>
15. Деловая онлайн-библиотека Alpina Digital <http://lib.alpinadigital.ru/>
16. Электронная библиотека Издательского дома «Гребенников» <https://grebennikon.ru/>
17. Математические журналы: полнотекстовая коллекция Математического института им. В.А. Стеклова РАН <https://www.mathnet.ru/>
18. Научная электронная библиотека eLibrary.ru <http://elibrary.ru>

- 19.Национальная электронная библиотека <http://нэб.рф/>
- 20.Ресурсы информационно-аналитического агентства по финансовым рынкам Cbonds.ru <https://cbonds.ru/>
- 21.СПАРК <https://spark-interfax.ru/>
- 22.CNKI. Academic Reference <https://ar.oversea.cnki.net/>
- 23.Электронные продукты издательства Elsevier <http://www.sciencedirect.com>
- 24.Emerald: Management eJournal Portfolio <https://www.emerald.com/insight/>
- 25.Реферативная база данных по математике MathSciNET <https://mathscinet.ams.org/mathscinet/>
- 26.Коллекция научных журналов Oxford University Press <https://academic.oup.com/journals/>
- 27.Электронные коллекции книг и журналов издательства Springer: <http://link.springer.com/>
- 28.Платформа STATISTA <https://www.statista.com/>
- 29.База данных научных журналов издательства Wiley <https://onlinelibrary.wiley.com/>
- 30.Каталог курсов Интернет Университета Информационных Технологий <http://www.intuit.ru/>.
- 31.The Python Tutorial // <https://docs.python.org/3/tutorial/index.html>.
- 32.The Python Standard Library // <https://docs.python.org/3/library/index.html>.
- 33.SciPy // <http://docs.scipy.org/doc/scipy/reference/>.
- 34.NumPy User Guide // <http://docs.scipy.org/doc/numpy/user/index.html>.
- 35.Система Thomson Reuters Eikon.
- 36.Бесплатный курс «Основы Python-разработки». <https://practicum.yandex.ru/python-free/>.
- 37.Программирование на Python. <https://stepik.org/course/67/promo>.

## **10. Методические указания для обучающихся по освоению дисциплины**

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

Практические занятия проводятся в соответствии с тематическим планом. При изложении материала рекомендуется использовать программный код из Jupyter Notebook и фрагменты печатных материалов по теме семинара. Материалы для занятий представлены в учебном пособии из списка основной литературы: «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. URL:<http://elib.fa.ru/fbook/books137296.pdf>. В ходе интерактивных занятий следует проводить разбор конкретных примеров программного кода из Jupyter Notebook.

Проведение практических занятий осуществляется в компьютерных классах и включает в себя реализацию всех этапов проектирования и реализации алгоритмов.

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

### **11.1. Комплект лицензионного программного обеспечения:**

1. Операционная система, пакет офисных программ.
2. Антивирус Kaspersky.
3. Дистрибутив Python Anaconda (свободно распространяемое ПО).

4. Браузер.
5. Файловый менеджер Far.

### ***11.2. 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/>.

### ***11.3. Сертифицированные программные и аппаратные средства защиты информации:***

- не используются.

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

Материально-техническая база Финансового университета, необходимая для осуществления образовательного процесса по данной дисциплине, включает в себя специальные помещения для проведения лекций, семинарских занятий, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации; помещения укомплектованы специализированной мебелью и техническими средствами обучения, необходимыми для представления информации большой аудитории.

Помещения для самостоятельной работы студентов включают в себя библиотеку с читальным залом, укомплектованную в соответствии с существующими нормами необходимой учебной и учебно-методической литературой и иными материалами; медиатеку с выходом в Интернет, компьютерные классы с возможностью работы в Интернет; аудитории для консультационной деятельности.