


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

Уфимский филиал Финуниверситета  
Кафедра «Экономика, менеджмент и маркетинг»

СОГЛАСОВАНО  
ООО «ХТЦ-УАН»  
Заместитель генерального директора  
 Р.Ф. Вагапов  
«01» \_\_\_\_\_ 2021 г.

МП

УТВЕРЖДАЮ  
Директор Уфимского филиала  
 Р.М. Сафуанов  
«02» \_\_\_\_\_ 2021 г.

**СОВРЕМЕННЫЕ ТЕХНОЛОГИИ ПРИКЛАДНОГО  
ПРОГРАММИРОВАНИЯ И ОБРАБОТКИ ДАННЫХ**

Рабочая программа дисциплины

для студентов, обучающихся по направлению подготовки

38.03.02 Менеджмент,

образовательные программы:

«Управление бизнесом» (Менеджмент и управление бизнесом)

Рекомендовано Ученым советом филиала  
(протокол № 35 от «21» \_\_\_\_\_ 08 2021 г.)

Одобрено кафедрой «Экономика, менеджмент и маркетинг»  
(протокол № 1 от «27» \_\_\_\_\_ 07 2021 г.)

Уфа 2021

**Автор - составитель канд. техн. наук, доцент кафедры «Математика и информатик» Исхаков З.Ф.**

## Оглавление

1. Наименование дисциплины .....	4
2. Перечень планируемых результатов освоения образовательной программы (перечень компетенций) с указанием индикаторов их достижения и планируемых результатов обучения по дисциплине .....	4
3. Место дисциплины в структуре образовательной программы .....	5
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.....	6
5.1. Содержание дисциплины .....	6
5.2. Учебно-тематический план .....	8
5.3. Содержание семинаров, практических занятий .....	9
6. Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине .....	10
6.1. Перечень вопросов, отводимых на самостоятельное освоение дисциплины, формы внеаудиторной самостоятельной работы.....	10
6.2. Перечень вопросов, заданий, тем для подготовки к текущему контролю (согласно таблице 2).....	10
7. Фонд оценочных средств для проведения промежуточной аттестации обучающихся по дисциплине .....	19
8. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины .....	25
9. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины .....	26
10. Методические указания для обучающихся по освоению дисциплины .....	27
11. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости) .....	27
11.1. Комплект лицензионного программного обеспечения: .....	27
11.2. Современные профессиональные базы данных и информационные справочные системы .....	27
Сертифицированные программные и аппаратные средства защиты информации ..	27
12. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине. ....	27

## 1. Наименование дисциплины

Современные технологии прикладного программирования и обработки данных

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

Таблица 1

Код компетенции	Наименование компетенции	Индикаторы достижения компетенции	Результаты обучения (владения, умения и знания), соотнесенные с компетенциями/индикаторами достижения компетенции
ПКН-2	Способность применять математические методы для решения стандартных профессиональных задач, интерпретировать полученные математические результаты	1. Демонстрирует знания математических методов, применяемых в менеджменте	<b>Знать:</b> математические методы, применяемых в менеджменте <b>Уметь:</b> применять математические методы, применяемые в менеджменте
		2. Применяет математические методы и модели для обоснования принятия управленческих решений	<b>Знать:</b> математические методы и модели для обоснования принятия управленческих решений <b>Уметь:</b> использовать математические методы и модели для обоснования принятия управленческих решений
		3. Содержательно интерпретирует результаты, полученные при использовании математических моделей	<b>Знать:</b> Знать связь предметной области и математического моделирования для интерпретации результатов, полученных при использовании математических моделей <b>Уметь:</b> интерпретировать результаты, полученные при использовании математических моделей
ПКП-4	Способность участвовать в разработке программ развития	1. Разрабатывать концепцию проекта, иерархическую структуру работ, календарно-ресурсный план и контроль за ходом программ развития организации	<b>Знать:</b> Методы проектирования проекта на концептуальном уровне, иерархическую структуру работ, календарно-ресурсный план и контроль за ходом программ развития организации <b>Уметь:</b> применять методы проектирования проекта на

	компаний, разработке обоснований проектов и управленческих решений, связанных с развитием бизнеса		концептуальном уровне, иерархическую структуру работ, календарноресурсный план и контроль за ходом программ развития организации
		2. Применяет современные модели развития и управления организацией	<b>Знать:</b> современные модели развития и управления организацией и возможности их реализации на языке Python <b>Уметь:</b> применить современные модели развития и управления организацией и реализовывать их на языке Python

### 3. Место дисциплины в структуре образовательной программы

Дисциплина относится к циклу профиля, дисциплина по выбору части образовательной программы «Управление бизнесом» (Менеджмент и управление бизнесом) по направлению подготовки 38.03.02 Менеджмент.

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

Общая трудоемкость дисциплины составляет 3 зачетных единиц.

Вид промежуточной аттестации - зачет

Вид текущего контроля – контрольная работа.

Вид учебной работы по дисциплине	Всего (в з/е и часах)	Семестр 7 (в часах)
<b>Общая трудоемкость дисциплины</b>	<b>3/108</b>	<b>108</b>
<b>Контактная работа-Аудиторные занятия</b>	<b>50</b>	<b>50</b>
<i>Лекции</i>	<i>16</i>	<i>16</i>
<i>Семинары, практические занятия</i>	<i>34</i>	<i>34</i>
<b>Самостоятельная работа</b>	<b>58</b>	<b>58</b>
Вид текущего контроля	Домашнее творческое задание	Домашнее творческое задание
Вид промежуточной аттестации	Зачет	Зачет

## **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 языке Python**

Задачи анализа данных, понятие набора данных (dataset). Подготовительные операции для выполнения анализа данных: загрузка данных, трансформация данных, изучение данных, очистка данных, визуализация данных.

Технологический стек анализа данных, построенный на базе языка программирования Python. Язык программирования Python: основные характеристики, возможности языка для решения задач анализа данных и машинного обучения. Версии языка программирования Python, дистрибутивы и библиотеки Python. Знакомство с дистрибутивом Anaconda и составом инструментов для задач анализа данных и машинного обучения, входящих в дистрибутив. Интерактивная оболочка IPython notebook: принципы работы и применение для решения задач анализа данных и машинного обучения.

#### **Тема 2. Основные синтаксические конструкции Python**

Знакомство с типами данных и операциями, переменными. Возможности работы со строками в Python. Основные операции над строками, функции и методы для работы со строками. Структура программы. Инструкции выражений, операторы сравнения, логические операторы. Инструкция ветвления if...else. Инструкция цикла while. Инструкция цикла for и Инструкции break, continue, pass, else.

Работа со списками в Python. Создание списка. Операции над списками. Перебор элементов списка. Многомерные списки. Методы списков. Кортежи. Работа со словарями в Python. Создание словаря. Операции над словарями. Перебор элементов словаря. Методы для работы со словарями. Множества.

#### **Тема 3. Базовые технологии для анализа данных**

Знакомство с библиотеками numpy и pandas и решением базовых задач подготовительных операций для выполнения анализа данных с помощью этих библиотек.

Постановки задач машинного обучения. Объекты и признаки. Типы признаков: бинарные, номинальные, порядковые, количественные. Типы задач машинного обучения: классификация, регрессия, прогнозирование, кластеризация. Примеры задач решаемых методами машинного обучения. Проблема недообучения / переобучения.

#### **Тема 4. Технологии работы со структурированными данными**

Обзор технологий хранения данных: файловых систем, реляционных СУБД, OLAP, Data Warehouses, не реляционных (“NoSQL”) баз данных. Сравнительный анализ и области применения различных технологий хранения информации. Работа с файлами. Работа с реляционными базами данных на примере SQLite.

Краткий обзор основных видов не реляционных баз данных: хранилищ «ключ-значение», хранилище семейств колонок, документо-ориентированная СУБД, баз данных на основе графов. Сравнительный анализ и области применения не реляционных баз данных.

Хранение и обмен структурированной информацией в виде документов или сообщений. Форматы представления переносимой структурированной информации. Сравнение различных принципов представления структурированной информации: закрытые и открытые форматы, бинарное и текстовое представление данных.

Универсальные форматы хранения структурированной информации (разметки документов): CSV, XML, HTML (XHTML), JSON. Язык разметки XML: основные принципы построения и специфика использования. Построение схемы документа с помощью XML DTD или XML Schema. HTML (XHTML) – отличие от XML, специфика использования. Формат представления структурированной информации JSON: принципы построения, специфика использования.

#### **Тема 5. Технологии обработки данных**

Знакомство с различными классами информационно-аналитических систем. Технологии Data Mining. Технологии анализа больших объемов данных (Big Data): причины возникновения, основные особенности функционирования и специфика создания приложений.

Сравнительный анализ различных подходов к анализу экономически значимой информации: от построения систем отчетов до алгоритмов машинного обучения. Особенности построения информационно-аналитических систем с применением алгоритмов машинного обучения. Основные этапы создания информационно-аналитических систем с использованием алгоритмов машинного обучения.

## 5.2. Учебно-тематический план

Таблица 2

№ п/п	Наименование тем (разделов) дисциплины	Трудоемкость в часах				Самостоятельная работа	Формы текущего контроля успеваемости
		Всего	Контактная работа - Аудиторная работа		Семинары, практические занятия		
			Общая	Лекции			
1	2	3	4	5	6	7	8
1.	Введение в программирован ие на языке Python	9	4	2	2	5	Устный опрос, проверка практических заданий
2.	Основные синтаксические конструкции Python	16	6	2	4	10	Устный опрос, проверка практических заданий
3.	Базовые технологии для анализа данных	27	12	4	8	15	Устный опрос, проверка практических заданий
4.	Технологии работы со структурированн ыми данными	22	12	4	8	10	Устный опрос, проверка практических заданий
5.	Технологии обработки данных	34	16	4	12	18	Устный опрос, проверка практических заданий
	В целом по дисциплине	108	50	16	34	58	Согласно учебному плану: Контрольная работа; Зачет

### 5.3. Содержание семинаров, практических занятий

Таблица 3

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
Тема 1. Введение в программирование на языке Python	Изучение технологического стека анализа данных, построенного на базе языка программирования Python. <b>Рекомендуемые источники из раздела 8: 1 – 5</b> <b>из раздела 9: 1-8, 10</b>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (30% времени на интерактивные технологии)
Тема 2. Основные синтаксические конструкции Python	Изучение базовых конструкций языка программирования Python. <b>Рекомендуемые источники из раздела 8: 1 – 5</b> <b>из раздела 9: 1-8, 10</b>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (33% времени на интерактивные технологии)
Тема 3. Базовые технологии для анализа данных	Знакомство с информационными технологиями анализа данных Python. <b>Рекомендуемые источники из раздела 8: 1 – 5</b> <b>из раздела 9: 1- 10</b>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (33% времени на интерактивные технологии)
Тема 4. Технологии работы со структурированными данными	Изучение примеров работы с форматами CSV, XML, XHTML, HTML, JSON при помощи библиотек на языке Python, проектирование собственного приложения, работающего с одним из данных форматов. <b>Рекомендуемые источники из раздела 8: 1 – 5</b> <b>из раздела 9: 1-10</b>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (25% времени на интерактивные технологии)
Тема 5. Технологии обработки данных	Изучение примеров построения аналитических инструментов на языке Python, проектирование инструментария анализа данных собственного приложения <b>Рекомендуемые источники из раздела 8: 1 – 5</b> <b>из раздела 9: 1-10</b>	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (25% 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

Таблица 4

Наименование тем (разделов) дисциплины	Перечень вопросов, отводимых на самостоятельное освоение	Формы внеаудиторной самостоятельной работы
Тема 1. Введение в программирование на языке Python	1. Знакомство с интерактивной оболочкой IPython notebook. 2. Изучение принципов работы в оболочке.	работа с литературой; работа с электронными источниками; разработка алгоритмов и программ.
Тема 2. Основные синтаксические конструкции Python	1. Работа с множествами, генераторы списков и словарей.	работа с литературой; работа с электронными источниками; разработка алгоритмов и программ.
Тема 3. Базовые технологии для анализа данных	1. Знакомство с библиотеками numpy и pandas и решением базовых задач подготовительных операций для выполнения анализа данных с помощью этих библиотек.	работа с литературой; работа с электронными источниками; разработка алгоритмов и программ.
Тема 4. Технологии работы со структурированными данными	1. Изучение библиотек Python для работ с данными в форматах XML, XHTML, HTML, JSON	работа с литературой; работа с электронными источниками; разработка алгоритмов и программ.
Тема 5. Технологии обработки данных	1. Изучение библиотек Python для анализа данных	работа с литературой; работа с электронными источниками; разработка алгоритмов и программ.

### 6.2. Перечень вопросов, заданий, тем для подготовки к текущему контролю (согласно таблице 2)

#### *Примеры заданий домашней творческой работы*

#### 1.Процедурное программирование

##### 1.1.Строки

1.1.1. Инвертировать последовательность слов, разделенных запятыми.

Пример: строка 'SIX, SEVEN, EIGHT, NINE, TEN' будет преобразована в: 'TEN, NINE, EIGHT, SEVEN, SIX'.

1.1.2. 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']]

1.1.3. 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'.

1.1.4. 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'

## 1.2.Генераторы

1.2.1. Используя генератор списков (и не используя код вне него) преобразовать строку по следующей логике: для каждого символа исходной строки создать в итоговом списке строку, содержащую копии символа в количестве, равном номеру символа в исходной строке. Пример: 'abcd' -> ['a', 'bb', 'ccc', 'dddd']

1.2.2. 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'}

1.2.3. Используя генератор словарей (и не используя код вне него) преобразовать словарь, в котором ключами являются кортежи из целых чисел в словарь, в котором ключом является среднее значение из чисел исходного ключа, значение оставить прежним.

Пример: {(2,4):'a', (1,1,1):'b', (2,3):'c'} -> {3.0:'a', 1.0:'b', 2.5:'c'}

1.2.4. Используя генератор списков (и не используя код вне него) преобразовать список кортежей в список кортежей по следующему правилу: если в кортеже четное количество элементов, то из него нужно удалить последний элемент. В остальных случаях кортежи оставить неизменными.

Пример: [(1,3,4), (2,1), (6,), (2,2,2,1)] -> [(1,3,4), (2,), (6,), (2,2,2,)]

1.2.5. Используя генератор списков (и не используя код вне него) преобразовать два списка (в первом содержатся целые числа, во втором строки, содержащие один символ) в словарь, в котором соответствующие друг другу пары значений из исходных списков преобразованы в целочисленный ключ и строку состоящую из повторенных символов (количество повторений равно значению ключа).

Пример [2, 4, 1, 3], ['a', 'b', 'c', 'd'] -> {2:'aa', 4:'bbbb', 1:'c', 3:'ddd'}

1.2.6. Используя генератор словарей (и не используя код вне него) преобразовать словарь, в котором ключами и значениями являются целые числа в список, в котором содержатся суммы исходных пар ключей и значений, причем, в список включаются только суммы, являющиеся четными числами.

Пример: {2:4, 3:2, 12:6, 5:4, 1:3} -> [6, 18, 4]

Используя генератор списков (и не используя код вне него) преобразовать список, содержащий положительные целые числа в список, элементами которого являются списки с длиной равной соответствующему числу в первом списке. Содержимым

вложенных списков являются последовательно идущие целые числа начиная с 1.  
Пример: [3, 1, 4] -> [[1, 2, 3], [1], [1, 2, 3, 4]]

1.2.7. Используя генератор списков (и не используя код вне него) преобразовать строку по следующей логике: для каждого символа исходной строки создать в итоговом списке строку, содержащую копии символа в количестве, равном номеру символа, рассчитанному с конца исходной строки.

Пример: 'abcd' -> ['aaaa', 'bbb', 'cc', 'd']

## **2.Объектно-ориентированное программирование**

### **2.1.Иерархия.**

2.1.1. Создать иерархию классов для фруктов, продающихся в магазине. Иерархия должна содержать не менее 5 классов, и не менее 3х уровней. Объекты должны содержать не менее 3х атрибутов и 2х методов (часть из которых должны быть перегружены). В конструкторах должны корректно использоваться конструкторы базовых классов.

Необходимо заполнить список представителями всех классов (всего не менее 10 объектов) и продемонстрировать работу полиморфизма.

2.1.2. 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.

2.1.3. 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.

## **3. Функции и функциональное программирование.**

### **3.1.Функции**

3.1.1. Реализовать функцию `summlate` для расчета накопленных сумм (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.

Пример: параметры: 1, 3, 2, 2 -> [1, 4, 6, 8]. Необязательный булевский параметр `mul` должен позволять заменять суммирование умножением.

Пример: параметры: 1, 3, 2, 2 -> [1, 3, 6, 12].

3.1.2. Реализовать функцию `gerl`, которая принимает на вход строку и набор заранее неизвестных параметров. Результатом функции является строка, в которой слова, совпадающие с именами параметров заменены на значения параметров.

Пример: строка: 'replace my val abc', параметры my='s1', abc='fff' -> Результат: 'replace s1 val fff'

3.1.3. Реализовать функцию `psort`, которая принимает на вход набор заранее неизвестных поименованных параметров. Функция возвращает список значений параметров, отсортированный по именам параметров.

Пример: `psort(c=21, a=22, ac=17, b=16)` -> `[22, 17, 16, 21]`

3.1.4. Реализовать функцию `psort`, которая принимает на вход набор заранее неизвестных поименованных параметров. Функция возвращает список имен параметров, отсортированный по значениям параметров.

Пример: `psort(c=21, a=22, ac=17, b=16)` -> `[b, ac, c, a]`

3.1.5. Реализовать функцию `nam_par`, которая принимает на вход заранее неизвестное количество параметров и необязательный параметр `name` в который можно передать строку. Функция возвращает словарь, в котором переданные параметры являются значениями, ключами для них являются соответствующие (сопоставленные по порядку следования) символы из строки `name`. Если строка `name` не задана, то значения присваиваются по порядку английского алфавита. Пример 1: `nam_par(7, 3, 1, 8, 10, 13, name='xyzafg')` -> `{'x':7, 'y':3, 'z':1, 'a':8, 'f':10, 'g':13}`

Пример 2: `nam_par(21, 'val', -3.5)` -> `{'a':21, 'b':'val', 'c':-3.5}`

3.1.6. Реализовать функцию `par_val`, которая принимает на вход заранее неизвестное количество именованных параметров (значения параметров - строки) и возвращает список имен параметров, которым соответствуют строки, содержащие более двух слов. Пример: `par_val(pp='abba war', fan='oneword', zr='a x')` -> `[pp, zr]`

#### 4.1. Рекурсии

4.1.1. Рекурсивно реализовать функцию `fib(n)` вычисляющую значение n-го числа Фибоначчи. Числа Фибоначчи — элементы числовой последовательности, в которой каждое последующее число равно сумме двух предыдущих чисел (Числа Фибоначчи: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, ...)

4.1.2. Создать рекурсивную реализацию функции поиска элемента в отсортированном списке методом деления пополам (бинарного поиска) `sort_search(sorted_list, value, from, to)`. Аргументы функции: `sorted_list` – отсортированный список, в котором проводится поиск; `value` – искомое значение; `from` – индекс элемента в списке, начиная с которого осуществляется поиск; `to` – индекс элемента в списке, до которого (не включительно) осуществляется поиск. Функция возвращает индекс первого вхождения `value` или `None`, в случае отсутствия элемента.

4.1.3. Создать рекурсивную реализацию функции поиска максимального числа в списке, содержащем целые числа. В качестве основного шага рекурсии использовать переход от поиска в данном списке к двум операциям поиска в списках вдвое меньшей длины.

Функция должна иметь вид `max_search(int_list, from, to)`. Аргументы функции: `int_list` – список, содержащий целые числа; `from` – индекс элемента в списке, начиная с

которого осуществляется поиск; `to` – индекс элемента в списке, до которого (не включительно) осуществляется поиск. Функция возвращает значение максимального элемента.

### 5.1. Декораторы

5.1.1. Реализовать декоратор, который выводит на печать возвращаемые значения функции.

5.1.2. Реализовать декоратор с именем `not_none`, который генерирует исключительную ситуацию если декорируемая функция вернула значения `None`.

5.1.3. Реализовать декоратор с именем `print_type`, выводящий на печать тип значения, возвращаемого декорируемой функцией.

### map/filter/reduce

5.1.4. При помощи механизма `map/filter/reduce` возвести в квадрат числа от 1 до 100, и рассчитать их сумму, не включая в сумму числа, кратные 10.

5.1.5. Дан список целых чисел. При помощи механизма `map/filter/reduce` рассчитать остаток от деления на 17 для каждого из чисел списка и получить произведение тех остатков, величина которых больше 7.

5.1.6. Дано предложение без знаков препинания. Превратить предложение в список слов. При помощи механизма `map/filter/reduce` отбросить у каждого слова последнюю букву и склеить в одну строку те обрезанные слова, длина которых больше 5.

## 4. Структуры данных

### 4.1. Стеки, очереди

4.1.1. При помощи стека (можно использовать любую реализацию стека) проверить что в строке содержащей открывающие и закрывающие скобки трех типов: `()`, `[]`, `{}`. Соблюдается корректный баланс и вложенность скобок.

4.1.2. Реализовать функцию `st_reverse(a_string)`, которая при помощи стека инвертирует строку (меняет порядок букв на обратный). Пример: `st_reverse('abcd') -> 'dcba'`

4.1.3. Реализовать функцию `list3_to2(list1, list2, list3)`, которая при помощи очереди превращает 3 списка одинаковой длины в 2 списка, длина которых не различается больше чем на 1, в полученных очередях должны чередоваться элементы из разных исходных списков и не должен нарушаться их порядок (т.е. если 'А' шло раньше 'В' в исходном списке, то 'В' не может идти раньше 'А' в итоговом списке).

## 5. Сортировки.

### 5.1. Простые сортировки

5.1.1. Реализовать алгоритм обменной сортировки.

5.1.2. Реализовать алгоритм сортировки выбором.

5.1.3. Реализовать алгоритм сортировки вставками.

5.1.4. Эффективные сортировки

5.1.5. Реализовать алгоритм сортировки Шелла.

5.1.6. Реализовать алгоритм быстрой сортировки.

### 5.1.7. Реализовать алгоритм сортировки слиянием.

#### ***Примерный перечень вопросов к домашней творческой работе***

1. Парадигмы программирования их суть и сильные стороны. Типичные представители различных парадигм, применение различных парадигм в Python.
2. Специфика типизации в языках программирования (различные аспекты типизации). Реализация типизации в Python.
3. Именованные переменные и другие объекты в Python (правила и соглашения). Числовые типы: литералы, объявление и операции.
4. Присвоение по ссылке и по значению. Специфика создания объектов и присвоения в Python, особенности работы с объектами целочисленного типа.
5. Разница между копированием и присвоением. Проблема утечки динамической памяти, сборка мусора. Копирование, присвоение и стратегия управления динамической памятью в Python.
6. Булевский тип, сравнения и условные операторы в Python.
7. Циклы в Python, работа и устройство цикла for, типичное применение range и enumerate в цикле for.
8. Строки в Python. Принципы работы и основные операторы, и функции.
9. Списки в Python. Различные способы создания и копирования списков в Python. Обход списка и поиск элементов в списке.
10. Списки в Python. Обращение к элементам списка и создание срезов. Стандартные агрегирующие функции, работающие со списками.
11. Списки в Python. Ключевые операции, проводящие к изменению списка и порождающие измененные списки.
12. Словари в Python. Основные способы создания, получения и изменения значений. Обход словарей.
13. Преобразование между словарями и списками в Python. Операции с представлениями словарей.
14. Операции со словарями, учитывающие возможное отсутствие ключа. Операции многоэлементного изменения словарей. Операции поэлементного извлечения из словаря и их использование.
15. Множества в Python. Основные способы создания, получения и изменения значений. Обход множеств.
16. Выполнение основных операций с парой множеств в Python.
17. Кортежи в Python. Отличия кортежей от списков. Распаковка и частичная распаковка кортежей.
18. Выражения генераторы и генераторы списков в Python. Использование условий в генераторах.
19. Генераторы множеств и словарей в Python. Использование условий в генераторах.

20. Функции стандартной библиотеки для работы с контейнерами.
21. 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.
22. 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.
23. Анонимные функции в Python их возможности и ограничения. Типичные сценарии использования анонимных функций.
24. Синтаксис и семантика обработки исключительных ситуаций в Python.
25. Создание пользовательских исключений и инструкция `assert`.
26. Базовые операции для работы с файлами в Python.
27. Использование инструкции `with ... as` на примере работы с файлами.
28. Использование модулей `pickle` и `shelve` для сохранения объектов в файл и их восстановления.
29. 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.
30. Импорт кода из пакетов. Организация пакетов в Python.
31. Концепция класса и объекта. Принципы и механизмы ООП.
32. Объявление класса, конструктор, создание объектов и одиночное наследование в Python.
33. Управление доступом к атрибутам класса в Python.
34. Полиморфизм и утиная типизация, и проверка принадлежности объекта к классу в языке Python.
35. Методы классов и статические переменные, и методы в Python.
36. Интроспекция и динамические операции с объектами в Python.
37. Специальные методы для использования пользовательских классов со стандартными операторами и функциями.
38. Основные возможности, поддерживаемые функциональными языками программирования. Поддержка функционального программирования в Python.
39. Концепция «функции – граждане первого класса» в языке программирования, поддержка этой концепции в Python. Специфика лямбда-функций в Python.
40. Глобальные и локальные переменные в функциях на примере Python. Побочные эффекты вызова функций и их последствия.
41. Вложенные функции и замыкания, специфика реализации в Python.
42. Функции высшего порядка и декораторы в Python.
43. Концепция `map/filter/reduce`. Работа `map()` в различных вариациях в Python.

44. Концепция map/filter/reduce. Работа filter() в Python. Использование модуля operator при работе с filter.
45. Концепция map/filter/reduce. Работа reduce() в Python. Использование reduce() с начальным значением, имеющим тип, отличный от возвращаемого редуцирующей функцией.
46. Итераторы в Python: встроенные итераторы, создание собственных итераторов, типичные способы обхода итераторов и принцип их работы.
47. Встроенные функции для работы с итераторами и возможности модуля itertools.
48. Функции генераторы и выражения генераторы: создание и применение в Python.
49. Специфика массивов, как структур данных. Динамические массивы – специфика работы, сложность операций. Специфика работа с array в Python.
50. Абстрактная структура данных стек: базовые и расширенные операции, их сложность. Реализация стека в Python.
51. Абстрактная структура данных очередь: базовые и расширенные операции, их сложность. Реализация очереди в Python.
52. Специфика реализации и скорости основных операций в очереди на базе массива и связанного списка.
53. Связанные списки: однонаправленные и двунаправленные. Сравнение скорости выполнения основных операций в связанных списках и в динамическом массиве.
54. Алгоритм сортировки выбором, сложность сортировки и возможности по ее улучшению.
55. Алгоритм сортировки вставками, его сложность. Алгоритм быстрого поиска в отсортированном массиве. Сложность поиска в отсортированном и не отсортированном массиве.
56. Алгоритм сортировки Шелла, сложность сортировки и возможности по ее улучшению.
57. Алгоритм быстрой сортировки, сложность сортировки и возможности по ее улучшению.
58. Алгоритм сортировки слиянием, сложность сортировки.

Критерии балльной оценки различных форм текущего контроля успеваемости содержатся в соответствующих методических рекомендациях кафедры.

**Рейтинговая система оценки знаний обучающихся по учебной дисциплине  
«Современные технологии прикладного программирования и обработки  
данных»**

Балльно-рейтинговая система представляет собой систему количественной оценки качества освоения образовательной программы высшего образования студентом в сравнении другим студентами.

В основу балльно-рейтинговой системы положена 100-балльная система оценки знаний студентов, используемая в качестве дополнения к официальной пятибалльной системе (семестровой, модульной) оценки знаний студентов, принятой в Российской Федерации, а также на основании Положения о проведении текущего контроля успеваемости и промежуточной аттестации обучающихся по программе бакалавриата и магистратуры в Финансовой университете от 23.03.2017 №0557/0

Критерии балльной оценки включают качество подготовки студентов к семинарским занятиям, выполнения различных видов самостоятельной работы, а также посещение аудиторных занятий. Балльная оценка текущего контроля успеваемости студента в семестре, составляет максимум 40 баллов.

Балльная оценка промежуточной аттестации составляет максимум 60 баллов. Общее количество баллов, которые может набрать студент – 100

Таблица 5

Балльной оценка текущего контроля успеваемости студента в семестре

№	Направление работы	Максимальное количество баллов
1	Посещение лекционных и практических занятий	6
2	Активная работа на практических занятиях	10
3	Защита контрольной работы.	20 и получение допуска к сдаче зачету
4	Научно-исследовательская работа студентов (участие в олимпиадах, международных и российских студенческих конференциях)	4

### Бальная оценка знаний на зачете

Максимальное количество баллов на зачете – 60

Количество баллов за ответы на зачете – максимально по 18 баллов на каждый вопрос.

Ответы на дополнительные вопросы – 6 баллов максимум.

Итоговая сумма баллов, полученная студентом на зачете и по результатам текущего контроля успеваемости в семестре, преобразуется в итоговое заключение в соответствии с Таблицей 6.

Таблица 6

- Порядок перевода 100-балльной оценки в итоговое заключение

50-100 баллов	Зачтено
Менее 50 баллов	Не зачтено

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

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

1. Найти максимальное значение результата применив метод покоординатного спуска
2. Реализовать оптимизационную задачу инструментами численной математики
3. Реализовать задачу линейного программирования на языке Python

4. Найти максимальное значение для целевой функции предметной области
5. Обосновать необходимость использования метода ЛП-поиска для поставленной задачи
6. Обосновать выбор математической модели для предложенной предметной задачи
7. Обосновать выбор жизненного цикла внедрения информационной системы
8. Разработать модель бизнес-процессов обследуемого предприятия
9. Произвести интеллектуальный анализ данных, составить прогноз изменения ключевого показателя
10. Создать прогнозную модель для решения задачи «что - если»

### **Примерные вопросы для подготовки к зачету**

1. Встроенные числовые типы языка Python.
2. Списки. Создание, основные операции.
3. Основные методы списка.
4. Кортежи. Создание, основные методы и операции.
5. Словари. Создание, основные операции. Методы для работы со словарями.
6. Множества. Создание, основные методы и операции.
7. Переменные. Правила именования переменных.
8. Динамическая типизация.
9. Операторы сравнения и логические операторы.
10. Инструкция if...else.
11. Инструкция цикла while.
12. Инструкция цикла for.
13. Создание и вызов функции.
14. Передача аргументов функцию.
15. Функции-генераторы.
16. Лямбда-функции.
17. Модули. Инструкции import и from.
18. Базовые принципы объектно-ориентированного программирования.
19. Класс, метод класса, атрибут класса. Определение класса и создание экземпляра класса.

20. Конструктор и деструктор.
21. Наследование.
22. Абстрактные методы класса.
24. Статические методы класса.
25. Свойства класса.
26. Исключения. Обработка исключений.
27. Пользовательские исключения.
28. Событие. Обработчик события. Цикл обработки событий.
29. Элемент Кнопка. Создание и настройка.
30. Элемент Кнопка. Создание обработчика события.
31. Элементы Надпись и Текстовое поле. Создание и настройка. Метод get().
32. Элемент Флажок. Создание, настройка, получение статуса флажка.
33. Элемент переключатель. Создание, настройка, доступ к значению.
34. Классы date, time и datetime.
35. Возможности библиотеки SymPy.
36. Возможности библиотеки NumPy.

### **Примеры тестовых заданий:**

#### **1. Какие ошибки здесь допущены?**

```
def factorial(n):
```

```
    if n == 0:
```

```
        return 1
```

```
    else:
```

```
        return n * factorial(n - 1)
```

```
print factorial(5)
```

- Функция не может вызывать сама себя
- В коде нет никаких ошибок
- Необходимо указать тип возвращаемого значения (**правильный ответ**)
- Функция всегда будет возвращать 1

#### **2. Имеется определение класса:**

```
class Ex:
```

```
    def __init__(self, x, y):
```

```

xy = x, y
self.position = xy
self._length = self.__len(x, y)
def __len(self, x, y):
    return abs(x) + abs(y)
def getlen(self):
    return self._length

```

```
p = Ex(1, 2)
```

Какой из вариантов его применения не допустим?

- `print p.getlen()`
- `print p.position`
- `print p.__len(1,2)` **(правильный ответ)**

### 3. Дан массив:

```
>>> c = array([[1,2], [2,3], [4,5]])
```

Чему равен срез `c[:,1]`?

- `array([1, 2, 4])`
- `*array([2, 3, 5])`
- `array([1, 2])`
- `array([2, 3])`

### 4. Как называется отношение, которое имеют следующие два класса:

```

class A(object):
    def __init__(self, x):
        self._mydata = x
    def m1(self):
        raise NotImplementedError

```

```

class B(A):
    def __init__(self, x):
        super(B, self).__init__(x)
    def m1(self):
        return self._mydata

```

- агрегация. Экземпляры А содержат экземпляры класса В
- наследование. В получается наследованием А **(правильный ответ)**
- ассоциация. Экземпляры А содержат ссылки на экземпляры класса В
- наследование. А получается наследованием В

### 5. Язык Python:

- имеет возможность интеграции с другими языками программирования **(правильный ответ)**
- является низкоуровневым языком программирования
- является высокоуровневым языком программирования **(правильный ответ)**
- имеет статическую типизацию
- имеет динамическую типизацию **(правильный ответ)**

**6. Привести переменную x к типу числа с плавающей точкой можно следующим способом:**

- (float)x
- x.float()
- x.\_\_float\_\_() **(правильный ответ)**
- float(x) **(правильный ответ)**
- x.\_\_class\_\_=float

**7. Из приведенных ниже высказываний укажите истинное:**

- перед использованием переменной она должна быть определена с указанием типа и инициализирована каким-либо значением
- перед использованием переменной она должна быть определена с указанием типа
- перед использованием переменной она должна быть инициализирована каким-либо значением **(правильный ответ)**
- переменные не обязаны быть инициализированы каким-либо значением перед использованием, а тип переменной определяется в зависимости от контекста
- перед использованием переменной она будет автоматически проинициализирована значением по умолчанию

**8. Укажите результат выполнения кода:**

```
x = {0:'P',1:'r',2:'i',3:'n',4:'t'}
```

```
x[0] = 'p'
```

```
print (x)
```

- {0: 'P', 1: 'r', 2: 'i', 3: 'n', 4: 't'}
- {0: 'p', 1: 'r', 2: 'i', 3: 'n', 4: 't'} **(правильный ответ)**
- {0: 'p'}
- скрипт не будет выполнен, так как содержит ошибки

**9. В языке Python существуют следующие операции для работы со списками:**

- объединение ( , )
- объединение ( + ) **(правильный ответ)**
- поиск различий ( - )
- тождественно равно ( == )
- меньше или равно ( <= ) **(правильный ответ)**

# 10. В языке Python существуют следующие операции сравнения:

- равно ( = )
- не равно (!=) (**правильный ответ**)
- много больше ( >> )
- меньше или равно ( <= ) (**правильный ответ**)
- больше или равно ( >= )

Таблица 5

Наименование компетенции	Наименование индикаторов достижения компетенции	Результаты обучения (умения и знания) соотнесенные с индикаторами достижения компетенции	Типовые контрольные задания
ПКН-2 Способность применять математические методы для решения стандартных задач, интерпретировать полученные математические результаты	1. Демонстрирует знания математических методов, применяемых в менеджменте	<b>Знать:</b> математические методы, применяемых в менеджменте <b>Уметь:</b> применять математические методы, применяемые в менеджменте	Заданий 1 Найти максимальное значение результата применив метод по координатного спуска  Задание 2 Реализовать оптимизационную задачу инструментами численной математики
	2. Применяет математические методы и модели для обоснования принятия управленческих решений	<b>Знать:</b> математические методы и модели для обоснования принятия управленческих решений <b>Уметь:</b> использовать математические методы и модели для обоснования принятия управленческих решений	Заданий 1 Реализовать задачу линейного программирования на языке Python  Задание 2 Найти максимальное значение для целевой функции предметной области
	3. Содержательно интерпретирует результаты, полученные при использовании математических моделей	<b>Знать:</b> Знать связь предметной области и математического моделирования для интерпретации результатов, полученных при использовании математических моделей <b>Уметь:</b> интерпретировать результаты, полученные при использовании математических моделей	Заданий 1 Обосновать необходимость использования метода ЛП-поиска для поставленной задачи  Задание 2 Обосновать выбор математической модели для предложенной предметной задачи
	1. Разрабатывать концепцию проекта, иерархическую структуру работ, календарно-ресурсный	<b>Знать:</b> Методы проектирования проекта на концептуальном уровне, иерархическую структуру работ,	Заданий 1 Обосновать выбор жизненного цикла внедрения информационной системы

ПКП-4 Способность участвовать в разработке программ развития компании, разработке обоснований проектов и управленческих решений, связанных с развитием бизнеса	план и контроль за ходом программ развития организации	календарноресурсный план и контроль за ходом программ развития организации <b>Уметь:</b> применять методы проектирования проекта на концептуальном уровне, иерархическую структуру работ, календарноресурсный план и контроль за ходом программ развития организации	Задание 2 Разработать модель бизнес-процессов обследуемого предприятия
	2. Применяет современные модели развития и управления организацией	<b>Знать:</b> современные модели развития и управления организацией и возможности их реализации на языке Python <b>Уметь:</b> применить современные модели развития и управления организацией и реализовывать их на языке Python	Заданий 1 Произвести интеллектуальный анализ данных, составить прогноз изменения ключевого показателя  Задание 2 Создать прогнозную модель для решения задачи «что -если»

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

### Основная литература

1. Белов, В. В. Алгоритмы и структуры данных : учебник / В. В. Белов, В. И. Чистякова. - Москва : КУРС : ИНФРА-М, 2020. - 240 с. - (Бакалавриат). - ISBN 978-5-906818-25-6. - Текст : электронный. - URL: <https://znanium.com/catalog/product/1057212>
2. Гуриков, С. Р. Основы алгоритмизации и программирования на Python : учебное пособие / С.Р. Гуриков. — Москва : ФОРУМ : ИНФРА-М, 2020. — 343 с. — (Высшее образование: Бакалавриат). - ISBN 978-5-00091-487-8. - Текст : электронный. - URL: <https://znanium.com/catalog/product/1206074>
3. Федоров, Д. Ю. Программирование на языке высокого уровня Python : учебное пособие для вузов / Д. Ю. Федоров. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2021. — 161 с. — URL: <https://ez.el.fa.ru:2428/bcode/472985>

### Дополнительная литература

4. Колдаев, В. Д. Структуры и алгоритмы обработки данных: учебное пособие / В. Д. Колдаев. - Москва : РИОР : ИНФРА-М, 2020. - 296 с. - (Высшее образование: Бакалавриат). - ISBN 978-5-369-01264-2. - Текст : электронный. - URL: <https://znanium.com/catalog/product/>
5. Чернышев, С. А. Основы программирования на Python : учебное пособие для вузов / С. А. Чернышев. — Москва : Издательство Юрайт, 2021. — 286 с. — URL: <https://ez.el.fa.ru:2428/bcode/477353>

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

1. PyIru 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. Официальный сайт продукта <https://www.python.org/>
  6. Информационно-образовательный портал Финансового университета при Правительстве Российской Федерации <http://portal.ufrf.ru/>
  7. Каталог курсов Интернет Университета Информационных Технологий <http://www.intuit.ru/>
  8. Электронная библиотека Финансового университета (ЭБ) <http://elib.fa.ru/> (<http://library.fa.ru/files/elibfa.pdf>)
  9. Scikit-learn Machine Learning in Python [Электронный ресурс]: сайт. – Режим доступа: <http://scikit-learn.org>
  10. Электронно-библиотечная система Znanium <http://www.znanium.com>
-

## 10. Методические указания для обучающихся по освоению дисциплины

Наименование методических материалов для обучающихся	Год утверждения	Местонахождение материала (ссылка на ИОП, информационный стенд кафедры/филиала, др.)
Методические указания к лекциям	2021	<a href="http://www.fa.ru/fil/ufa/about/ums/Pages/info.aspx">http://www.fa.ru/fil/ufa/about/ums/Pages/info.aspx</a>
Методические указания к практическим занятиям	2021	<a href="http://www.fa.ru/fil/ufa/about/ums/Pages/info.aspx">http://www.fa.ru/fil/ufa/about/ums/Pages/info.aspx</a>
Методические указания самостоятельной работе	2021	<a href="http://www.fa.ru/fil/ufa/about/ums/Pages/info.aspx">http://www.fa.ru/fil/ufa/about/ums/Pages/info.aspx</a>
Методические указания по формам текущего контроля успеваемости	2021	<a href="http://www.fa.ru/fil/ufa/about/ums/Pages/info.aspx">http://www.fa.ru/fil/ufa/about/ums/Pages/info.aspx</a>

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

### 11.1. Комплект лицензионного программного обеспечения:

Продукты компании Microsoft, включая ОС Windows и Office.

### 11.2. Современные профессиональные базы данных и информационные справочные системы

Электронное периодическое издание Справочная Правовая Система Консультант Бюджетные организации: версия Проф.

## Сертифицированные программные и аппаратные средства защиты информации

Сертифицированные программные и аппаратные средства защиты информации – не используются.

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

Учебная аудитория для проведения всех видов занятий, предусмотренных программой бакалавриата, оснащенная оборудованием и техническими средствами обучения.