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

**Департамент анализа данных, принятия решений
и финансовых технологий**

Макрушин С.В.

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ ПРИКЛАДНОГО ПРОГРАММИРОВАНИЯ И ОБРАБОТКИ ДАННЫХ

Рабочая программа дисциплины

для студентов, обучающихся
по направлениям подготовки 01.03.02 «Прикладная математика и информатика», 09.03.03 «Прикладная информатика», 38.03.04 «Государственное и муниципальное управление», 38.03.05 «Бизнес-информатика», 39.03.01 «Социология», 41.03.04 «Политология», 42.03.01 «Реклама и связи с общественностью», 38.03.01 «Экономика», 38.03.02 «Менеджмент», 38.03.03 «Управление персоналом», 40.03.01 «Юриспруденция», 10.03.01 «Информационная безопасность», 43.03.02 «Туризм»

Москва 2017

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

**Департамент анализа данных, принятия решений
и финансовых технологий**

УТВЕРЖДАЮ
Проректор по развитию
образовательных программ
и международной деятельности
_____ Е.А. Каменева
28.11.2017 г.

Макрушин С.В.

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ ПРИКЛАДНОГО ПРОГРАММИРОВАНИЯ И ОБРАБОТКИ ДАННЫХ

Рабочая программа дисциплины

для студентов, обучающихся
по направлениям подготовки 01.03.02 «Прикладная математика и информатика», 09.03.03 «Прикладная информатика», 38.03.04 «Государственное и муниципальное управление», 38.03.05 «Бизнес-информатика», 39.03.01 «Социология», 41.03.04 «Политология», 42.03.01 «Реклама и связи с общественностью», 38.03.01 «Экономика», 38.03.02 «Менеджмент», 38.03.03 «Управление персоналом», 40.03.01 «Юриспруденция», 10.03.01 «Информационная безопасность», 43.03.02 «Туризм»

*Рекомендовано Ученым советом факультета
«Прикладная математика и информационные технологии»*
(*протокол № 48 от 21 ноября 2017 г.*)

*Одобрено Департаментом анализа данных, принятия решений и финансовых технологий
(протокол № 4 от 21 ноября 2017 г.)*

Москва 2017

Рецензент: **Л.Н. Чернышов** –к.ф-м.н., доцент департамента анализа данных, принятия решений и финансовых технологий

Макрушин С.В. «Современные технологии прикладного программирования и обработки данных». Рабочая программа дисциплины по выбору университетского блока. – М.: Финансовый университет, 2017. – 43 с.

Дисциплина «Современные технологии прикладного программирования и обработки данных» является дисциплиной по выбору университетского блока дисциплин по выбору.

Дисциплина «Современные технологии прикладного программирования и обработки данных» знакомит с возможностями использования современного ИТ-инструментария для анализа данных, в том числе в финансово-экономических приложениях.

Рабочая программа содержит требования к уровню освоения содержания дисциплины, объем дисциплины и виды учебной работы, программу дисциплины и тематику практических занятий, учебно-методическое и информационное обеспечение.

Учебное издание

Сергей Вячеславович Макрушин

Современные технологии прикладного программирования и обработки данных

Рабочая программа дисциплины

Компьютерный набор, верстка С.В. Макрушин
Формат 60x90/16. Гарнитура *Times New Roman*

Усл. п.л. 1,2. Изд. № - 2017. Тираж - экз.

Заказ № _____

Отпечатано в Финансовом университете

Содержание

1. Наименование дисциплины

Современные технологии прикладного программирования и обработки данных

2. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

Структура планируемых результатов обучения по дисциплине

Для направлений подготовки: Прикладная математика и информатика, Прикладная информатика, Государственное и муниципальное управление, Бизнес-информатика, Социология, Политология, Реклама и связи с общественностью:

ОК-1 – способностью использовать основы философских знаний для формирования мировоззренческой позиции		
Знания	***Умения***	***Владения***
Знать: • основы философских знаний; • концепцию мировоззренческой позиции.	Уметь: • использовать философские знания; • формировать мировоззренческую позицию.	Владеть • навыками использования знания для формирования мировоззренческой позиции.
ОК-7 – способностью к самоорганизации и самообразованию		
Знания	***Умения***	***Владения***
Знать: • принципы самоорганизации; • принципы самообразования.	Уметь: • заниматься самоорганизацией; • заниматься самообразованием.	Владеть • навыками самоорганизации; • навыками самообразования.

Для направлений подготовки: «Экономика», «Менеджмент», «Управление персоналом», «Юриспруденция»:

ОНК-1 – Способность использовать основные научные законы в профессиональной деятельности		
Знания	***Умения***	***Владения***
Знать: • основные научные законы; • принципы профессиональной деятельности.	Уметь: • использовать основные научные законы; • применять научные законы для профессиональной деятельности.	Владеть • навыком использования основных научных законов в профессиональной деятельности.
ОНК-2 – Владение культурой мышления, способность к восприятию, анализу и мировоззренческой оценке происходящих процессов и закономерностей		
Знания	***Умения***	***Владения***
Знать: • принципы культуры мышления; • принципы восприятия, анализа и мировоззренческой оценки; • о современных процессах и закономерностях.	Уметь: • воспринимать; • анализировать; • проводить мировоззренческую оценку.	Владеть • культурой мышления; • способностью к восприятию происходящих процессов и закономерностей; • анализу и мировоззренческой оценке происходящих процессов и закономерностей.

Для направления подготовки: «Информационная безопасность»:

ОК-1 – Способность использовать основы философских знаний для формирования мировоззренческой позиции		
Знания	***Умения***	***Владения***
Знать: • основы философских знаний; • концепцию мировоззренческой позиции.	Уметь: • использовать философские знания; • формировать мировоззренческую позицию.	Владеть • способностью использовать знания для формирования мировоззренческой позиции.
ОК-8 – Способность к самоорганизации и самообразованию		
Знания	***Умения***	***Владения***
Знать:	Уметь:	Владеть

• принципы самоорганизации; • принципы самообразования.	• заниматься самоорганизацией; • заниматься самообразованием.	• навыками самоорганизации; • навыками самообразования.

Для направления подготовки: Туризм:

ОК - 1 – способностью использовать основы философских знаний, анализировать главные этапы и закономерности исторического развития для создания социальной значимости своей деятельности		
Знания	***Умения***	***Владения***
Знать: • основы философских знаний; • главные этапы и закономерности исторического развития; • принципы социальной значимости деятельности.	Уметь: • использовать основы философских знаний; • анализировать главные этапы и закономерности исторического развития.	Владеть • навыками использования основ философских знаний для создания социальной значимости своей деятельности; • навыками использования анализа главных этапов и закономерностей исторического развития.
ОК - 5 – Способность к самоорганизации и самообразованию		
Знания	***Умения***	***Владения***
Знать: • принципы самоорганизации; • принципы самообразования.	Уметь: • заниматься самоорганизацией; • заниматься самообразованием.	Владеть • навыками самоорганизации; • навыками самообразования.

3. Место дисциплины в структуре образовательной программы

Дисциплина «Современные технологии прикладного программирования и обработки данных» является дисциплиной по выбору студента университетского блока.

Для изучения дисциплины «Современные технологии прикладного программирования и обработки данных» необходимы знания, умения и компетенции, которые получены при изучении школьных курсов математики и информатики.

4. Объем дисциплины

Общая трудоёмкость дисциплины составляет 3 зачётных единицы.

Форма промежуточной аттестации – зачет.

Форма текущего контроля – отсутствует.

Вид учебной работы по дисциплине	Всего (в з/е и часах)	Семестр 3 или 4 или 5 (в часах)
Общая трудоемкость дисциплины	**3/108**	**108**
Аудиторные занятия	**36**	**36**
Лекции	18	18
Практические и семинарские занятия, в т.ч.	18	18
занятия в интерактивной форме	*18 (50%)*	*18 (50%)*
Самостоятельная работа	**72**	**72**
Вид текущего контроля	-	-
Вид промежуточной аттестации	Зачет	Зачет

5. Содержание дисциплины

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

№ п/п	Наименование темы дисциплины	Трудоёмкость в часах						Формы текущего контроля успеваемости
		Всего	Аудиторная работа				Самостоятельная работа	
			Общая	Лекции	Практические и семинарские занятия	Занятия в интерактивных формах		
1.	Введение в программирование на языке Python	23	8	4	4	4	15	УО, ППЗ
2.	Основные синтактические конструкции Python	23	8	4	4	4	15	УО, ППЗ
3.	Базовые технологии для анализа данных	16	4	2	2	2	12	УО, ППЗ, КР
4.	Технологии работы со структурированными данными	23	8	4	4	4	15	УО, ППЗ
5.	Технологии обработки данных	23	8	4	4	4	15	УО, ППЗ
	ИТОГО:	**108**	**36**	**18**	**18**	**18/ 50 %**	**72**	

*Сокращения в таблице: **УО** – устный опрос; **ППЗ** – проверка практических заданий, **КР** – домашняя контрольная работа

5.3. Содержание практических и семинарских занятий

Целью проведения практических занятий является приобретение студентами практических навыков анализа финансово-экономических данных на языке Python.

Темы практических занятий приведены в табл. 5, 6.

В качестве интерактивной формы обучения используется метод мозгового штурма, дерево решений (для выбора наилучшего варианта), обсуждение в группах (группа из двух студентов).

Таблица 4

Наименование темы (раздела) дисциплины	Тематика практических и/ или семинарских занятий	Содержание практических и/ или семинарских занятий	Формы проведения занятий (с указанием % занятий, проводимых в интерактивной форме)	Вопросы для самостоятельной работы студентов	Рекомендуемые источники из разделов 8,9
Тема 1. Введение в программирование на языке Python	Установка Python, установка дистрибутива Anaconda. Работа в интерактивном режиме интерпретатора. Интерактивная оболочка IPython notebook: принципы работы и применения. Среда программирования. Использование документации.	Входной контроль. Изучение технологического стека анализа данных, построенного на базе языка программирования Python.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (30% времени на интерактивные технологии)	Знакомство с интерактивной оболочкой IPython notebook. Изучение принципов работы в оболочке.	8.1; 8.2; 8.4; 8.3
Тема 2. Основные синтактические конструкции Python	Изучение базовых конструкций языка программирования Python	Входной контроль. Изучение базовых конструкций языка программирования Python	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (33% времени на интерактивные технологии)	Работа с множествами, генераторы списков и словарей.	8.1; 8.2; 8.4; 8.3
Тема 3. Базовые технологии для анализа данных	Знакомство с информационными технологиями анализа данных	Входной контроль. Знакомство с информационными технологиями анализа данных Python.	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (33% времени на интерактивные технологии)	Знакомство с библиотеками numpy и pandas и решением базовых задач подготовительных операций для выполнения анализа данных с	8.5; 8.8; 8.9; 8.10; 8.11; 8.12

				помощью этих библиотек.	
Тема 4. Технологии работы со структурированными данными	Создание на языке Python приложений использующих универсальные форматы хранения структурированной информации	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	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (25% времени на интерактивные технологии)	Создание на языке программирования Python приложения работающего со структурированной информацией в одном из следующих форматов: XML, XHTML, HTML, JSON	8.2; 8.4; 8.3; 8.5; 8.8; 8.9; 8.10; 8.11; 8.12
Тема 5. Технологии обработки данных	Знакомство с аналитическими инструментами на языке Python	Изучение примеров построения аналитических инструментов на языке Python, проектирование инструментария анализа данных собственного приложения	Индивидуальное выполнение заданий, групповой разбор результатов выполнения заданий (25% времени на интерактивные технологии)	Создание на языке программирования Python инструментария анализа данных для собственного приложения	8.2; 8.4; 8.3; 8.5; 8.8; 8.9; 8.10; 8.11; 8.12

6. Учебно-методическое обеспечение самостоятельной работы.

6.1. Формы внеаудиторной самостоятельной работы.

№ п/п	НТ*	ФВСТ	Т/Ч	Указание тем, отводимых для самостоятельного освоения обучающимся
1	2	3	4	5
1.	Знакомство с современным стеком ИТ анализа данных	РЛ, РЭИ, РАП	15	Знакомство с интерактивной оболочкой IPython notebook. Изучение принципов работы в оболочке.
2.	Основные синтактические конструкции Python	РЛ, РЭИ, РАП	15	Работа с множествами, генераторы списков и словарей.
3.	Базовые технологии для анализа данных	РЛ, РЭИ, РАП	12	Знакомство с библиотеками numpy и pandas и решением базо-

№ п/п	НТ*	ФВСТ	Т/Ч	Указание тем, отводимых для самостоятельного освоения обучающимся
				вых задач подготовительных операций для выполнения анализа данных с помощью этих библиотек.
4.	Технологии работы со структурированными данными	РЛ, РЭИ, РАП	15	Изучение библиотек Python для работ с данными в форматах XML, XHTML, HTML, JSON
5.	Технологии продвинутого анализа данных	РЛ, РЭИ, РАП	15	Изучение библиотек Python для анализа данных
ИТОГО:			**72**	

* Сокращения в таблице: **НТ** – Названия тем; **ФВСТ** – Формы внеаудиторной самостоятельной работы; **Т/Ч** – Трудоемкость в часах; **РЛ** – работа с литературой; **РЭИ** – работа с электронными источниками; **РАП** – разработка алгоритмов и программ.

6.2. Методическое обеспечение для аудиторной и внеаудиторной самостоятельной работы.

Тематика внеаудиторной самостоятельной работы: контрольной работы (КР), соответствует содержанию дисциплины, и определяется преподавателем.

Примерная тема контрольного домашней работы «Освоение синтаксических и семантических конструкций языка программирования Python».

Для выполнения домашней контрольной работы необходимо повторить материалы лекций и семинаров. Результат контрольной работы должен в согласованные сроки быть передан преподавателю в электронной форме. Перед решением обязательно должно быть размещено условие задачи.

Тексты лекций, задания на самостоятельную работу, примеры решения типовых задач, а также вспомогательные материалы находятся на преподавательском диске, доступном студентам.

Для проведения тестирования используется комплекс тестовых заданий, разработанных для каждой темы. Результаты тестирования, устных опросов, контрольных работ и т.д. фиксируются в ведомостях преподавателя, которые находятся на преподавательском диске.

Для каждой темы разработаны дополнительные задачи, решение которых в инициативном порядке могут выполнять студенты. Преподаватель за каждые N самостоятельно разработанных программ может выставлять дополнительно M баллов.

Рабочая версия интегрированной среды программирования, используемая при разработке программ, а также соответствующая документация находятся в открытом доступе в сети Интернет и доступны для скачивания, инсталляции и использования в соответствии с принятой процедурой регистрации.

Примерный вариант заданий контрольной 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 словарая, его значениями и наоборото. Значения, которые в исходном словаре повторяются не добавлять в итоговы 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 ключем 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 неизменнымии.

Приемер: [(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 повтарений 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]

1.2.7. Используя генератор списков (и не используя код вне него) преобразовать список содержащий положительные целые числа в список, элементами которого являются списки с длинной 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]]

1.2.8. 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 cummulate для рассчета 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 repl , 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 соответсвующие (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]

1.1. Рекурсии

1.1.1. Рекурсивно реализовать функцию fib(n) вычисляющую значение n-го числа Фибоначи. 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, …).

1.1.2. 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.

1.1.3. 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.

1.2.Декораторы

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

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

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

1.3. map/filter/reduce

1.3.1. 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.

1.3.2. 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.

1.3.3. 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.

2. Структуры данных

2.1. Стеки, очереди

2.1.1. При помощи стека (можно исопльзовать любую реализацию стека) проверить что в строке содержащей открывающие и закрывающие скобки трех типов: (), [], {}.

Соблюдается корректный баланс и вложенность скобок.

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

2.1.3. 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 'A' шло раньше 'B' в исходном списке, то 'B' не может идти раньше 'A' в итоговом списке.

3. Сортировки.

3.1. Простые сортировки

3.1.1. Реализовать алгоритм обменной сортировки.

3.1.2. Реализовать алгоритм сортировки выбором.

3.1.3. Реализовать алгоритм сортировки вставками.

3.2. Эффективные сортировки

3.2.1. Реализовать алгоритм сортировки Шелла.

3.2.2. Реализовать алгоритм быстрой сортировки.

3.2.3. Реализовать алгоритм сортировки слиянием.

Примеры теоретических заданий домашней контрольной:

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. Именование переменных и других объектов 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 вариация в Python.

44. Концепция map/filter/reduce. Работа filter() в Python. Использования 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.

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

7.1. Перечень компетенций, формируемых в процессе освоения дисциплины

Перечень компетенций, формируемых в процессе освоения дисциплины содержится в разделе 2.

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

Для направлений подготовки: Прикладная математика и информатика, Прикладная информатика, Государственное и муниципальное управление,

Бизнес-информатика, Социология, Политология, Реклама и связи с общественностью:

ОК-1 – способностью использовать основы философских знаний для формирования мировоззренческой позиции.

Оценка уровня форсированности компетенции

Показатели Оценивания	*Критерии оценивания компетенций*	*Шкала оценивания*
Знать: • основы философских знаний; • концепцию мировоззренческой позиции. **Уметь:** • использовать философские знания; • формировать мировоззренческую позицию. **Владеть:** • навыками использования знания для формирования мировоззренческой позиции.	**Знать:** • основы философских знаний; • концепцию мировоззренческой позиции.	*Пороговый уровень*
	Знать: • основы философских знаний; • концепцию мировоззренческой позиции. **Уметь:** • использовать философские знания; • формировать мировоззренческую позицию.	*Продвинутый уровень*
	Знать: • основы философских знаний; • концепцию мировоззренческой позиции. **Уметь:** • использовать философские знания; • формировать мировоззренческую позицию. **Владеть:** • навыками использования знания для формирования мировоззренческой позиции.	*Высокий уровень*

ОК-7 – способностью к самоорганизации и самообразованию

Показатели Оценивания	*Критерии оценивания компетенций*	*Шкала оценивания*
Знать: • принципы самоорганизации; • принципы самообразования. **Уметь:** • заниматься самоорганизацией; • заниматься самообразованием. **Владеть:** • навыками самоорганизации; • навыками самообразования.	**Знать:** • принципы самоорганизации; • принципы самообразования.	*Пороговый уровень*
	Знать: • принципы самоорганизации; • принципы самообразования. **Уметь:** • заниматься самоорганизацией; • заниматься самообразованием.	*Продвинутый уровень*
	Знать: • принципы самоорганизации; • принципы самообразования. **Уметь:** • заниматься самоорганизацией; • заниматься самообразованием. **Владеть:** • навыками самоорганизации; навыками самообразования.	*Высокий уровень*

Для направлений подготовки: «Экономика», «Менеджмент», «Управление персоналом», «Юриспруденция»:

ОНК-1 – Способность использовать основные научные законы в профессиональной деятельности

Показатели Оценивания	*Критерии оценивания компетенций*	*Шкала оценивания*

Знать: • основные научные законы; • принципы профессиональной деятельности. **Уметь:** • использовать основные научные законы; • применять научные законы для профессиональной деятельности. **Владеть:** • навыком использования основных научных законов в профессиональной деятельности.	**Знать** • ключевые технологии работы с данными с использованием языка программирования Python; • подходы к обработке данных с использованием языка программирования Python.	***Пороговый уровень***
	Знать • ключевые технологии работы с данными с использованием языка программирования Python; • подходы к обработке данных с использованием языка программирования Python. **Уметь** • создавать собственные функции, классы, модули; • создавать приложения для анализа данных. **Владеть** • навыками использования интегрированных сред на базе языка программирования Python для анализа данных и построения математических моделей.	***Продвинутый уровень***
	Знать: • основные научные законы; • принципы профессиональной деятельности. **Уметь:** • использовать основные научные законы; • применять научные законы для профессиональной деятельности. **Владеть:** • навыком использования основных научных законов в профессиональной деятельности.	***Высокий уровень***

ОНК-2 – Владение культурой мышления, способность к восприятию, анализу и мировоззренческой оценке происходящих процессов и закономерностей

Показатели Оценивания	***Критерии оценивания компетенций***	***Шкала оценивания***

Знать: • принципы культуры мышления; • принципы восприятия, анализа и мировоззренческой оценки; • о современных процессах и закономерностях. **Уметь:** • воспринимать; • анализировать; проводить мировоззренческую оценку. **Владеть:** • культурой мышления; • способностью к восприятию происходящих процессов и закономерностей; • анализу и мировоззренческой оценке происходящих процессов и закономерностей.	**Знать:** • принципы культуры мышления; • принципы восприятия, анализа и мировоззренческой оценки; • о современных процессах и закономерностях.	***Пороговый уровень***
	Знать: • принципы культуры мышления; • принципы восприятия, анализа и мировоззренческой оценки; • о современных процессах и закономерностях. **Уметь:** • воспринимать; • анализировать; проводить мировоззренческую оценку.	***Продвинутый уровень***
	Знать: • принципы культуры мышления; • принципы восприятия, анализа и мировоззренческой оценки; • о современных процессах и закономерностях. **Уметь:** • воспринимать; • анализировать; проводить мировоззренческую оценку. **Владеть:** • культурой мышления; • способностью к восприятию происходящих процессов и закономерностей; анализу и мировоззренческой оценке происходящих процессов и закономерностей.	***Высокий уровень***

Для направления подготовки: «Информационная безопасность»:

ОК-1 – Способность использовать основы философских знаний для формирования мировоззренческой позиции

Показатели	*Критерии оценивания*	*Шкала*

Оценивания	*компетенций*	*оценивания*
Знать: • основы философских знаний; • концепцию мировоззренческой позиции. **Уметь:** • использовать философские знания; • формировать мировоззренческую позицию. **Владеть:** • способностью использовать знания для формирования мировоззренческой позиции.	**Знать:** • основы философских знаний; • концепцию мировоззренческой позиции.	***Пороговый уровень***
	Знать: • основы философских знаний; • концепцию мировоззренческой позиции. **Уметь:** • использовать философские знания; • формировать мировоззренческую позицию.	***Продвинутый уровень***
	Знать: • основы философских знаний; • концепцию мировоззренческой позиции. **Уметь:** • использовать философские знания; • формировать мировоззренческую позицию. **Владеть:** • способностью использовать знания для формирования мировоззренческой позиции.	***Высокий уровень***

ОК-8 – Способность к самоорганизации и самообразованию

Показатели Оценивания	*Критерии оценивания компетенций*	*Шкала оценивания*
Знать: • принципы самоорганизации; • принципы самообразования. **Уметь:**	**Знать:** • принципы самоорганизации; • принципы самообразования.	***Пороговый уровень***

• заниматься самоорганизацией; • заниматься самообразованием. **Владеть:** • навыками самоорганизации; • навыками самообразования.	**Знать:** • принципы самоорганизации; • принципы самообразования. **Уметь:** • заниматься самоорганизацией; • заниматься самообразованием.	***Продвинутый уровень***
	Знать: • принципы самоорганизации; • принципы самообразования. **Уметь:** • заниматься самоорганизацией; • заниматься самообразованием. **Владеть:** • навыками самоорганизации; • навыками самообразования.	***Высокий уровень***

Для направления подготовки: Туризм:

ОК - 1 – способностью использовать основы философских знаний, анализировать главные этапы и закономерности исторического развития для создания социальной значимости своей деятельности

Показатели Оценивания	***Критерии оценивания компетенций***	***Шкала оценивания***
Знать: • основы философских знаний;	**Знать:** • основы философских знаний; • главные этапы и закономерности исторического развития; • принципы социальной значимости деятельности.	***Пороговый уровень***

• главные этапы и закономерности исторического развития; • принципы социальной значимости деятельности. **Уметь:** • использовать основы философских знаний; • анализировать главные этапы и закономерности исторического развития. **Владеть:** • навыками использования основ философских знаний для создания социальной значимости своей деятельности; • навыками использования анализа главных этапов и закономерностей исторического развития.	**Знать:** • основы философских знаний; • главные этапы и закономерности исторического развития; • принципы социальной значимости деятельности. **Уметь:** • использовать основы философских знаний; • анализировать главные этапы и закономерности исторического развития.	***Продвинутый уровень***
	Знать: • основы философских знаний; • главные этапы и закономерности исторического развития; • принципы социальной значимости деятельности. **Уметь:** • использовать основы философских знаний; • анализировать главные этапы и закономерности исторического развития. **Владеть:** • навыками использования основ философских знаний для создания социальной значимости своей деятельности; • навыками использования анализа главных этапов и закономерностей исторического развития.	***Высокий уровень***

ОК - 5 – Способность к самоорганизации и самообразованию

Показатели Оценивания	***Критерии оценивания компетенций***	***Шкала оценивания***

Знать: • принципы самоорганизации; • принципы самообразования. **Уметь:** • заниматься самоорганизацией; • заниматься самообразованием. **Владеть:** • навыками самоорганизации; • навыками самообразования.	**Знать:** • принципы самоорганизации; • принципы самообразования.	***Пороговый уровень***
	Знать: • принципы самоорганизации; • принципы самообразования. **Уметь:** • заниматься самоорганизацией; • заниматься самообразованием.	***Продвинутый уровень***
	Знать: • принципы самоорганизации; • принципы самообразования. **Уметь:** • заниматься самоорганизацией; • заниматься самообразованием. **Владеть:** • навыками самоорганизации; • навыками самообразования.	***Высокий уровень***

Промежуточная аттестация по дисциплине проводится в форме зачета (3 или 5 семестр).

7.3. Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, владений.

Примеры вопросов 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.

6. Методы для работы со словарями.
7. Множества. Создание, основные методы и операции.
8. Переменные. Правила именования переменных.
9. Динамическая типизация.
10. Операторы сравнения и логические операторы.
11. Инструкция if...else.
12. Инструкция цикла while.
13. Инструкция цикла for.
14. Создание и вызов функции.
15. Передача аргументов функцию.
16. Функции-генераторы.
17. Лямбда-функции.
18. Модули. Инструкции import и from.
19. Базовые принципы объектно-ориентированного программирования.
20. Класс, метод класса, атрибут класса. Определение класса и создание экземпляра класса.
21. Конструктор и деструктор.
22. Наследование.
23. 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.

7.4. Методические материалы, определяющие процедуры оценивания знаний, умений и владений.

Соответствующие приказы, распоряжения ректората о контроле уровня освоения дисциплин и сформированности компетенций студентов.

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

Основная литература

1. Колдаев В.Д. Основы алгоритмизации и программирования [Электронный ресурс]: учебное пособие / В.Д. Колдаев; под ред. Л.Г. Гагариной. – Москва: ИД ФОРУМ: ИНФРА-М, 2012, 2015, 2017. – 416 с. – Режим доступа: http://znanium.com/go.php?id=902236

2. Колдаев В.Д. Структуры и алгоритмы обработки данных [Электронный ресурс]: учебное пособие / В.Д. Колдаев. – Москва: ИЦ РИОР: НИЦ ИНФРА-М, 2014. – 296 с. – Режим доступа: http://znanium.com/catalog.php?bookinfo=418290

Дополнительная литература

3. Прохоренок Н.А. Python. Самое необходимое / Н.А. Прохоренок. – Санкт-Петербург: БХВ-Петербург, 2011. – 416 с.

4. Прохоренок Н.А. Python и PyQt. Разработка приложений / Н.А. Прохоренок. – Санкт-Петербург: БХВ-Петербург, 2012.

5. Маккинли У. Python и анализ данных / У. Маккинли; пер. с англ. А.А. Слинкин. – Москва: ДМК Пресс, 2015..

6. Прохоренок Н.А. Python 3. Самое необходимое / Н.А. Прохоренок, В.А. Дронов. – Санкт-Петербург: БХВ-Петербург, 2016. – 464 с.

7. Доусон М. Программируем на Python / М. Доусон; пер. с англ. – Санкт-Петербург: Питер, 2015. – 416 с.

8. Лутц М. Изучаем Python, / М. Лутц; пер. с англ. – 4-е издание. – Санкт-Петербург: Символ-Плюс, 2011. – 1280 с.

9. Лутц М. Программирование на Python, том I / М. Лутц; пер. с англ.– 4-е издание. — Санкт-Петербург: Символ-Плюс, 2011. – 992 с.

10. Лутц М. Программирование на Python, том II / М. Лутц; пер. с англ.– 4-е издание. – Санкт-Петербург: Символ-Плюс, 2011. – 992 с.

11. SciPy // http://docs.scipy.org/doc/scipy/reference/

12. NumPy User Guide // http://docs.scipy.org/doc/numpy/user/index.html

13. The Python Standard Library //

9. Перечень ресурсов сети «Интернет»

9.1. Pylru 1.0.9 [Электронный ресурс]: сайт. – Режим доступа: https://pypi.python.org/pypi/pylru

9.2. Python Data Analysis Library [Электронный ресурс]: сайт. – Режим доступа: http://pandas.pydata.org/

9.3. Python Documentation [Электронный ресурс]: сайт. – Режим доступа: http://python.org/doc/

9.4. Python Standard Library [Электронный ресурс]: сайт. – Режим доступа: https://docs.python.org/2/library/

9.5. Scikit-learn Machine Learning in Python [Электронный ресурс]: сайт. – Режим доступа: http://scikit-learn.org

9.6. Официальный сайт продукта https://www.python.org/

9.7. Информационно-образовательный портал Финансового университета при Правительстве Российской Федерации http://portal.ufrf.ru/

9.8. Каталог курсов Интернет Университета Информационных Технологий http://www.intuit.ru/

10. Электронная библиотека Финансового университета (ЭБ) http://elib.fa.ru/ (http://library.fa.ru/files/elibfa.pdf)
11. Электронно-библиотечная система Znanium http://www.znanium.com

10. Методические указания по освоению дисциплины.

При изложении лекции используется проблемный подход, что значительно расширяет предоставленный материал. На преподавательском диске находятся тексты лекций, материалы практических занятий, разбитых по темам. Там же приведены постановки задач, образцы программ решения типовых задач и справочные материалы.

Для получения доступа к облачному хранилищу студенты должны получить соответствующую ссылку от преподавателя.

При переходе к новой теме проводится тестирование, направленное на оценивание теоретических знаний. Помимо тестирования, может проводиться выборочный устный опрос студентов.

Практические навыки оцениваются путем разработки прикладных программ. Студенты должны самостоятельно и вовремя решать поставленные преподавателем задачи. Преподаватель должен отмечать и поощрять наиболее исполнительных студентов.

Примеры практических заданий для самостоятельного выполнения студентами

1. Инвертировать последовательность слов, разделенных запятыми. Пример: строка 'SIX, SEVEN, EIGHT, NINE, TEN' будет преобразована в: 'TEN, NINE, EIGHT, SEVEN, SIX'.
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']]
3. 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']
4. Реализовать функцию cummulate для рассчета 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]
5. Рекурсивно реализовать функцию fib(n) вычисляющую значение n-го числа Фибоначи. 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, …).
6. Реализовать декоратор, который выводит на печать возвращаемые значения функции.

7. При помощи стека (можно исопльзовать любую реализацию стека) проверить что в строке содержащей открывающие и закрывающие скобки трех типов: (), [], {}. Соблюдается корректный баланс и вложенность скобок
8. Реализовать функцию st_reverse(a_string), которая при помощи стека инвертирует строку (меняет порядок букв на обратный). Пример: st_reverse('abcd') -> 'dcba'
9. 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.
10. Реализовать алгоритм обменной сортировки.

11. Используемые информационные технологии.

11.1. Для обеспечения взаимодействия преподавателя и студентов используются облачные технологии. Всем студентам преподаватель предоставляет доступ на чтение пространства преподавательского диска.

11.2. Для хранения своих материалов каждому студенту предоставляется сетевая папка. Однако в такой папке нельзя хранить исполняемые файлы, поэтому проекты необходимо сохранять в виде архива.

11.3. Преподавателю также предоставляется сетевая папка tasks. Студентам эта папка доступна только для чтения.

11.4. При проведении зачета в рамках вычислительной сети преподаватель может использовать комплекс программ, разработанных на кафедре, для блокирования помощи третьих лиц и автоматического сбора результатов зачета.

12. Материально-техническая база образовательного процесса.

12.1. Для проведения лекций и практических занятий необходима аудитория, оснащенная проектором и компьютерами с постоянным подключением к сети Интернет.

На компьютеры преподавателя и студентов должно быть установлено следующее программное обеспечение:

1. Операционная система Windows 7 и выше.
2. Браузер Google Chrom.
3. Дистрибутив языка Python 3.4 (или более поздней версии) Anaconda 3
4. Для манипулирования с файлами файловый менеджер Far
5. Архиватор.
6. Пакет MS Office.

12.2. При чтении лекций, проведении семинаров и практических занятий необходима возможность подключения личного ноутбука преподавателя к проектору через интерфейсы VGA или HDMI.