Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)

Институт дистанционного образования

УТВЕРЖДЕНО:
Директор ИДО - проректор по РДО
М.О. Шепель

Оценочные материалы по дисциплине

Программирование на Python

по направлению подготовки

09.04.03 Прикладная информатика

Направленность (профиль) подготовки:
«Науки о данных»

Форма обучения
Очная

Квалификация
Магистр

Год приема
2024

СОГЛАСОВАНО:
Руководитель ОПОП
Д.Д. Даммер

Председатель УМК
С.Б. Велединская

Томск – 2024

1. Компетенции и индикаторы их достижения, проверяемые данными оценочными материалами

Целью освоения дисциплины является формирование следующих компетенций:

ОПК-1 - способность самостоятельно приобретать, развивать и применять математические, естественнонаучные, социально-экономические и профессиональные знания для решения нестандартных задач, в том числе в новой или незнакомой среде и в междисциплинарном контексте;

ОПК-2 - способность разрабатывать оригинальные алгоритмы и программные средства, в том числе с использованием современных интеллектуальных технологий, для решения профессиональных задач;

ОПК-7 - способность использовать методы научных исследований и математического моделирования в области проектирования и управления информационными системами;

Результатами освоения дисциплины являются следующие индикаторы достижения компетенций:

ИОПК-1.3 Знает основы развития и применения математических, естественнонаучных, социально-экономических и профессиональных знаний для решения задач;

ИОПК-2.1 Знает основы необходимых методов алгоритмизации и программирования для решения профессиональных задач;

ИОПК-7.2 Знает основы применения полученных знаний при решении задач профессиональной деятельности;

2. Оценочные материалы текущего контроля и критерии оценивания

№	**Этапы формирования компетенций (разделы дисциплины/модуля/практики)**	**Код и наименование результатов обучения**	**Вид оценочного средства** (тесты, задания, кейсы, вопросы и др.)
	Тема 1. Функции Введение в программирование на Python. Подробнее о функциях. Проверка аргументов и аргументы по умолчанию. Продвинутая передача аргументов. Продвинутая обработка аргументов. Lambda-функции	ИОПК-2.1; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
	Тема 2. Продвинутое использование функций в Python Вложенные функции. Область видимости переменных. Изменение переменных вне области видимости. Вложенные функции и области видимости. Практика. Рекурсия. Встроенная функция map(). Встроенная функция filter()	ИОПК-2.1; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
	Тема 3. Практика (HW) Основы Python. Погружение в типы данных. Условные операторы. Циклы. Функции. Тематический проект. Задачи с собеседований (HW).	ИОПК-2.1; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект

	Тема 4. Инструменты для Data Science (HW) Интегрированная среда разработки (IDE). Базовые функции VS Code. Jupyter Notebook. Работа с GitHub. Воспроизводимость кода. Работа с Google Colab. Задачи с собеседований (HW).	ИОПК-1.3; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
	Тема 5. Библиотека NumPy Модуль Collections. Counter и defaultdict. Deque и OrderedDict. Закрепление знаний. Модуль NumPy. Типы данных. Массивы. Действия с массивами. Операции с векторами. Случайные числа.	ИОПК-2.1; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
	Тема 6. Введение в Pandas Pandas.Series. Pandas.DataFrame. Работа с различными источниками данных в Pandas. Знакомимся с данными: недвижимость. Исследование структуры DataFrame. Статистические методы. Фильтрация данных в DataFrame.	ИОПК-2.1; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
	Тема 7. Базовые приёмы работы с данными в Pandas Базовые операции со столбцами DataFrame. Работа с датами в DataFrame. Создание и преобразование столбцов с помощью функций. Тип данных Category.	ИОПК-2.1; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
	Тема 8. Продвинутые методы работы с данными в Pandas Сортировка данных в DataFrame. Погружение в типы данных. Группировка данных в DataFrame. Сводные таблицы. Объединение DataFrame: знакомимся с новыми данными. Объединение DataFrame: concat. Объединение DataFrame: join, merge.	ИОПК-2.1; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
	Тема 9. Визуализация данных Обзор типов визуализации. Знакомимся с новыми данными: коронавирус. Графические возможности библиотеки Pandas. Графические возможности библиотеки Matplotlib. Графические возможности библиотеки Seaborn. Графические	ИОПК-1.3; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект

	возможности библиотеки Plotly. Искусство визуализации.		
	Тема 10. Очистка данных Знакомство с новыми данными: данные о квартирах от Сбера. Работа с пропусками: как их обнаружить? Работа с пропусками: методы обработки. Выбросы: почему появляются и чем опасны? Методы выявления выбросов. Работа с дубликатами и неинформативными признаками.	ИОПК-1.3; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
	Тема 11. Как выгружать данные из файлов разных форматов Работа с текстовыми файлами. Работа с файлами Excel. JSON. Что это? JSON. Открываем JSON-файл и извлекаем данные. JSON. Работаем с pandas. Из JSON в pandas. JSON. Работаем с pandas. Из pandas в JSON. XML. Что это. XML. Контент XML-файла. XML. Загружаем, создаем, сохраняем.	ИОПК-1.3; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
	Тема 12. Как получать данные из веб-источников и API Веб-запросы. Библиотека requests. Парсинг сайтов. Библиотека BeautifulSoup. Работа с API. Как настроить регулярную выгрузку данных. Задачи с собеседований (HW).	ИОПК-1.3; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
13.	Тема 13. Принципы ООП в Python и отладка кода Принципы ООП. Объекты и классы. Атрибуты и методы. Практические примеры. Проверка знаний. Применение ООП для работы с файлами. Исключения. Тонкости обработки исключений. Собственные классы исключений.	ИОПК-2.1; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект
14.	Тема 14. Markdown и GIT для создания портфолио Язык разметки Markdown. Синтаксис Markdown. Системы контроля версий. Git и GitHub. Git. Основные операции. Git. Игнорирование и работа с удалённым репозиторием. Git. Ветвление и конфликты. Методологии ветвления. Культура	ИОПК-1.3; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект

	коммитов. Форк. Рекомендации к составлению портфолио на GitHub		
15.	Тема 15. Анализ резюме из HeadHunter (PJ) Постановка задачи. Исследование структуры данных. Преобразование данных. Исследование зависимостей в данных. Очистка данных. Итоги и финальное задание (PJ)	ИОПК-1.3; ИОПК-7.2	Тест Контрольная работа Домашнее задание Проект

Примеры элементов текущего контроля:

Тест (ИОПК-2.1)
Задание 1
Какой оператор позволяет оставить функцию пустой?

1. break
2. pass
3. return
4. yield

Задание 2
Выберите верное утверждение об операторе return:

1. Позволяет вернуть только одно значение
2. Используется для печати результата на экран
3. Завершает выполнение кода функции
4. Завершает выполнение исходного кода основного скрипта

Задание 3
Выберите верное утверждение об использовании аргументов по умолчанию:

1. Аргументы по умолчанию могут быть указаны в сигнатуре функции только после обязательных
2. В качестве аргументов по умолчанию корректно использовать изменяемые типы данных
3. Значение аргумента по умолчанию невозможно задать при запуске функции
4. Если при запуске функции не передать значения аргументов по умолчанию, возникнет ошибка

Задание 4
Для чего в сигнатуре функции используется конструкция *args, **kwargs?
Например, def my_func(*args, **kwargs): ...

1. Чтобы записать все порядковые аргументы в kwargs, а именованные — в args
2. Чтобы записать все порядковые аргументы в args, а именованные — в kwargs
3. Чтобы распаковать значения из списка в несколько переменных
4. Чтобы указать фиксированное число входных параметров

Задание 5.
Выберите вариант, где lambda-функция создана правильным образом:

1. lambda x: return len(x)
2. lambda x, y; x*y
3. lambda x+y: x,y
4. lambda *args: sum(args)/len(args)

Критерии оценивания: тест считается пройденным, если обучающий ответил правильно как минимум на половину вопросов.

Домашнее задание 1 (ИОПК-2.1, ИОПК-7.2)
Ниже представлен код, в котором используется несколько переменных.
Обратите внимание, что мы специально создали запутанные имена переменных.

```
inner_counter = 0
 def outer_function(value, x):
   global_counter = value
    def inner_function(x):
      global inner_counter
      nonlocal global_counter
      inner_counter += 1
      global_counter += 1
      result = x + inner_counter + global_counter
      return result
    return inner_function(x)
local_result = outer_function(value=3, x=4)
```

Домашнее задание 2 (ИОПК-2.1, ИОПК-7.2)
Мы реализовали функцию processing_list(). Она работает следующим образом:

- на вход подаётся список answers, состоящий из строк;
- внутри себя функция обрабатывает каждую строку: приводит её к нижнему регистру и удаляет из неё специальные стоп-символы.

Процедура предобработки каждой строки вынесена в отдельную функцию preprocessing_answer().

```
# Внешняя функция для обработки списка из строк
def processing_list(answers):
   # Внутренняя функция для выполнения обработки одного элемента списка
   def preprocessing_answer(answer):
      # Приводим к нижнему регистру
      answer = answer.lower()
      # Удаляем стоп-символы
      stop_symbols = '"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~!'
      # Создаём цикл по символам в строке
      for sym in stop_symbols:
         # Если стоп-символ есть в строке answer
         if sym in answer:
            # Заменяем стоп-символы в строке на пустые строки
            answer = answer.replace(sym, '')
      return answer
   # Создаём пустой список, куда будем добавлять результаты
   result = []
   # Создаеё цикл по всем элементам списка
   for answer in answers:
       # Добавляем предобработанную строку в лог-список
      result.append(preprocessing_answer(answer))
   return result
```

Ответьте на несколько вопросов по представленному коду.

1. Какая функция является объемлющей? Введите её имя в поле для ответа.
2. Что будет выведено на экран в результате выполнения следующего кода:

print(processing_list(['Мама мыла раму, раму мыла мама', 'Hello & world!']))

1. ['Мама мыла раму, раму мыла мама', 'Hello & world!']
2. ['мама мыла раму раму мыла мама', 'hello world']
3. ['мама мыла раму раму мыла мама', 'hello world!']
4. Будет выведена ошибка NameError

Проект №1 (ИОПК-1.3)

УСЛОВИЕ ЗАДАЧИ

Создаём некоторое приложение, предусматривающее регистрацию пользователей.
Пока идет реализация небольшого функционала для регистрации. Для этого необходимо написать функцию register(surname, name, date, middle_name, registry).
Она будет иметь следующие аргументы:

surname — фамилия пользователя;
name — имя пользователя;
date — дата рождения пользователя в виде строки формата "DD.MM.YYYY" (например, "13.01.2001");
middle_name — отчество пользователя;
registry — список, в который необходимо добавить полученные аргументы в виде кортежа. Порядок следующий: фамилия, имя, отчество, день, месяц, год рождения.
Регистрацию будем имитировать добавлением данных о пользователях в список в виде кортежа. Функция должна возвращать список, в который добавила запись.

Примеры вызовов функции:

```
reg = register('Petrova', 'Maria', '13.03.2003', 'Ivanovna')
reg = register('Ivanov', 'Sergej', '24.09.1995', registry=reg)
reg = register('Smith', 'John', '13.02.2003', registry=reg)
print(reg)
```

[('Petrova', 'Maria', 'Ivanovna', 13, 3, 2003), ('Ivanov', 'Sergej', None, 24, 9, 1995), ('Smith', 'John', None, 13, 2, 2003)]
Примечание. Значение отчества по умолчанию должно быть None, так как отчество может быть не у всех регистрирующихся.

Значения дня, месяца и года рождения должны быть представлены в виде целых чисел.

Также нужно сделать так, чтобы пустой список создавался в том случае, если он не был передан извне. То есть по умолчанию registry имеет значение None, и если при вызове функции список так и не был передан, то он создаётся в теле функции.

Контрольная работа (ИОПК-2.1, ИОПК-7.2.)
Контрольная работа состоит из 4 теоретических вопросов и 6 задач.

Задание 1

Какие типы данных в Python относятся к числовым?

Отметьте все подходящие варианты ответов.

1. int
2. str
3. float
4. list

Задание 2

Дан код, в котором определяется значение переменной a:

a = '123' + 1

Однако сейчас этот код выдаёт ошибку:

Какие из перечисленных ниже способов помогут исправить код так, чтобы он отработал без ошибок?

1. Привести число 1 к строковому типу данных с помощью функции str().
2. Никак, этот код нельзя исправить.
3. Привести строку '123' к числовому типу данных с помощью функции int().
4. Изменить имя переменной.

Задание 3

С помощью какого метода можно перевести все символы в строке в верхний регистр?

1. upper()
2. lower()
3. capitalize()
4. split()

Задание 4

Дана строка:

python_string = 'Hello! My name is Python. I will help you to analyze some data.'

Какой из вариантов кода позволит получить подстроку 'Python' из исходной строки?

1. python_string[17:24].
2. python_string[18:25].
3.python_string[18:24]
4. python_string[17:25].

Задание 5

Дана строка python_string.

Посчитайте количество символов в этой строке и возведите получившееся число в третью степень. Результат запишите в переменную result.

Примеры работы программы:

python_string = 'Hello! My name is Python. I will help you to analyze some data.'

result = 250047

python_string = 'Привет! Меня зовут Питон. Я помогу вам проанализировать некоторые данные.'

result = 389017

Примечание. Обратите внимание, что для отправки кода на проверку переменную python_string объявлять не нужно. Не забудьте удалить строку с её объявлением перед отправкой кода на тестирование.

Задание 6

Представим, что мы разрабатываем алгоритм прогнозирования времени доставки еды. Наш алгоритм принимает на вход информацию о географических координатах клиента, расположении ресторана, загруженности на дорогах и так далее. Выходом алгоритма является примерное время в минутах, которое необходимо курьеру на доставку.

Мы хотим проверить адекватность работы алгоритма — для этого будем измерять разницу (ошибку) между спрогнозированным и реальным временем доставки. Условимся, что знак ошибки (в меньшую или большую сторону мы ошибаемся) нас не интересует — нужно только абсолютное значение.

Прогнозируемое время доставки хранится в переменной predicted_time, реальное время — в переменной real_time.

Найдите модуль разницы между предсказанным и истинным значениями времени, результат округлите до целого и сохраните в переменную absolute_error.

Пример работы программы:
predicted_time = 67.321
real_time = 59.839
absolute_error = 7
predicted_time = 38.873
real_time = 26.124
absolute_error = 13

Примечание. Обратите внимание, что для отправки кода на проверку переменные predicted_time и real_time объявлять не нужно. Не забудьте удалить строку с их объявлением перед отправкой кода на тестирование.

Задание 7

Дана строка input_string. Избавьтесь от знаков препинания в ней и напишите программу для подсчёта количества слов в этой строке. Программа должна записывать результат в переменную count_words.

Гарантируется, что из знаков препинания в строке input_string могут содержаться только точки ('.'), запятые (','), восклицательные ('!') и вопросительные ('?') знаки.

Примеры работы программы:
input_string = 'Hello! My name is Python. I will help you to analyze some data.'
count_words = 13

input_string = 'There are many great articles about Artificial Intelligence and its benefits for business and society. However, many of these articles are too technical for the average reader.'
count_words = 27

Задание 8

Мы обучаем нейронную сеть классифицировать изображения. Для обучения необходимо узнать формат файла с изображением: JPG/PNG/GIF и т. д. Это связано с тем, что от формата входных данных зависит процесс предобработки изображений (например, GIF-изображение необходимо предварительно разложить на кадры).

В переменной file_path задан полный путь до файла с изображением. Найдите его расширение, результат занесите в переменную file_extension.

Также найдите имя файла (без учёта расширения), результат занесите в переменную file_name.

Гарантируется, что разделителем между папками при указании пути является символ '/'.

Примеры работы программы:

file_path = 'data/images/train/10394.jpg'
file_name = '10394'
file_extension = 'jpg'

file_path = 'data/images/validation/748923.gif'
file_name = '748923'
file_extension = 'gif'

file_path = 'data/images/384300.png'
file_name = '384300'
file_extension = 'png'

Примечание. Обратите внимание, что для отправки кода на проверку переменную file_path объявлять не нужно. Не забудьте удалить строку с её объявлением перед отправкой кода на тестирование.

Задание 9

Предположим, мы обучили нейронную сеть генерировать текст (предложения), однако на этапе тестирования выяснилось, что мы допустили ошибку при обучении и слова генерируются в неправильной последовательности — справа налево, а не слева направо.

Ошибку мы нашли, однако оказалось, что переобучать модель очень долго и дорого. Поэтому мы решили вручную разворачивать сгенерированный текст на этапе постобработки. Он хранится в переменной generated_text в виде строки.

Создайте новую строку updated_text, в которой слова будут храниться в обратном порядке.

Для простоты гарантируется, что в предложениях, сгенерированных моделью, нет знаков препинания и весь текст представлен в нижнем регистре.

Примеры работы программы:
generated_text = "глаза нее на поднял он и она попросила что-нибудь скажи"
updated_text = "скажи что-нибудь попросила она и он поднял на нее глаза"

generated_text = "задачи своей решения способ или информацию ищет он поисковик в запрос вводит человек, когда"
updated_text = когда человек вводит запрос в поисковик он ищет информацию или способ решения своей задачи

Примечание. Обратите внимание, что для отправки кода на проверку переменную generated_text объявлять не нужно. Не забудьте удалить строку с её объявлением перед отправкой кода на тестирование.

Задание 10

В приложении, которое мы разрабатываем, предусмотрена функция смены пароля. Нам необходимо записывать в базу данных событий информацию о том, что пользователь сменил пароль.

Напишите функцию change_password(), которая должна возвращать отформатированную строку в следующем виде: "Пользователь {user_name} сменил пароль на {new_password}".

Здесь:

user_name — имя пользователя,
new_password — новый пароль.

Мы уже сделали заготовку функции — вам осталось только задать строку. Переменные user_name и new_password являются аргументами функции change_password().

Запишите форматированную строку вместо знаков вопроса.

```
def change_password(user_name, new_password):
   return ???
```

Примеры работы программы:

```
print(change_password('Andrey', '31andrey12QK'))
## Пользователь Andrey сменил пароль на 31andrey12QK
print(change_password('Vasilisk', 'uaf12laK'))
## Пользователь Vasilisk сменил пароль на uaf12laK
```

Критерии оценивания:

Результаты контрольной работы определяются оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Оценка «отлично» выставляется, если даны правильные ответы на все теоретические вопросы и все задачи решены без ошибок.

Оценка «хорошо» выставляется, если даны правильные ответы на не менее чем 3 из 4 теоретических вопросов, и допущено не более 1-2 незначительных ошибок в решении задач (не более 1 ошибки в каждой задаче).

Оценка «удовлетворительно» выставляется, если даны правильные ответы на не менее чем 2 из 4 теоретических вопросов, и допущено не более 3-4 незначительных ошибок в решении задач (не более 2 ошибок в каждой задаче).

Оценка «неудовлетворительно» выставляется, если даны правильные ответы менее чем на 2 теоретических вопроса и допущено более 4 ошибок в решении задач.

Примеры ошибок:

Незначительные ошибки:

- Ошибки синтаксиса, которые легко исправить (например, забытый двоеточие в конце строки).
- Неправильное использование методов или функций, не влияющее на общий результат.
- Мелкие логические ошибки, которые можно быстро исправить.

Значительные ошибки:

- Неправильное понимание задания, приводящее к неверному решению.
- Ошибки, существенно влияющие на результат выполнения задачи.
- Пропущенные важные шаги в решении задач.

Дополнительные критерии:

Отличное выполнение должно демонстрировать полное и глубокое понимание материала.

Хорошее выполнение должно показывать уверенное знание материала с небольшими недочетами.

Удовлетворительное выполнение показывает базовое понимание и способность решать основные задачи.

Неудовлетворительное выполнение указывает на необходимость дополнительного изучения материала и исправления пробелов в знаниях.

3. Оценочные материалы итогового контроля (промежуточной аттестации) и критерии оценивания

Промежуточная аттестация проводится в формате итогового индивидуального проекта. Индивидуальный проект представляет из себя письменный отчет (файл расширения docx/doc/pdf) на основе проведенного исследования, в рамках которого самостоятельно ставится исследовательский вопрос, выбираются данные, оценивается модель.

Отчет состоит из двух частей. Структура отчета следующая.

Часть 1

1. постановка задачи
2. ход исследования
3. обзор имеющегося опыта
4. формулировка ожидаемых результатов
5. данные и их подготовка
6. выбранная эконометрическая модель

Часть 2

1. описание данных, в т.ч. описательная статистика
2. оценка моделей, проведение тестов
3. интерпретация результатов
4. выводы

На основании оценок и комментариев преподавателя по первой части отчета, выполняется вторая часть отчета с учетом замечаний. Подробные требования к каждому пункту обозначаются преподавателем в презентации на первом занятии.

Экзамен оценивается по результатам текущего контроля – выполнение промежуточных тестов по темам курса (11 тестов= 66 баллам) и по результатам индивидуального итогового индивидуального проекта (34 балла).

Критерии оценивания итогового проекта:

Часть 1 (19 баллов)

1. постановка задачи (3)
2. ход исследования (2)
3. обзор имеющегося опыта (4)
4. формулировка ожидаемых результатов (2)
5. данные и их подготовка (4)
6. выбранная эконометрическая модель (4)

Часть 2 (15 баллов)

1. описание данных, в т.ч. описательная статистика (4)
2. оценка моделей, проведение тестов (5)
3. интерпретация результатов (3)
4. выводы (3)

На основании оценок и комментариев преподавателя по первой части отчета, выполняется вторая часть отчета с учетом замечаний.

Итоговая оценка складывается из суммы баллов, набранных при прохождении тестовых заданий, предусмотренных для текущего контроля и баллов, набранных при выполнение итогового проекта:

Итоговая сумма набранных баллов	Оценка
≤ 40	неудовлетворительно
от 41 до 60	удовлетворительно
от 61 до 80	хорошо
от 81 до 100	отлично

4. Оценочные материалы для проверки остаточных знаний (сформированности компетенций)

Тестовые вопросы:

1. Какая функция используется для проверки типа аргументов в функции? (ИОПК-1.3)
   a) isinstance()
   b) type()
   c) eval()
   d) assert()

2. Как в Python можно реализовать вложенную функцию? (ИОПК-2.1)
   a) Определить функцию внутри другой функции
   b) Использовать функцию lambda
   c) Использовать декораторы
   d) Определить функцию вне другой функции

3. Какие функции из библиотеки NumPy используются для создания массивов? (ИОПК-2.1)
   a) array(), arange(), linspace()
   b) dataframe(), series(), concat()
   c) plot(), scatter(), bar()
   d) read_csv(), read_excel(), read_json()

4. Какой метод Pandas используется для фильтрации данных в DataFrame? (ИОПК-1.3)
   a) sort()
   b) filter()
   c) query()
   d) loc[]

5. Какой библиотекой в Python можно использовать для веб-запросов? (ИОПК-2.1)
   a) requests
   b) numpy
   c) pandas
   d) json

Задачи:

Задача 1 (ИОПК-1.3):

Дано: Массив данных о ценах на недвижимость prices = [250000, 300000, 350000, 400000, 450000]

Требуется: Напишите функцию, которая принимает массив цен и возвращает массив цен, увеличенных на 10%.

Решение:

python

```
def increase_prices(prices):
    return [price * 1.1 for price in prices]

prices = [250000, 300000, 350000, 400000, 450000]
new_prices = increase_prices(prices)
print(new_prices)  # Output: [275000.0, 330000.0, 385000.0, 440000.0, 495000.0]
```

Задача 2 (ИОПК-2.1):

Дано: Два массива arr1 = [1, 2, 3] и arr2 = [4, 5, 6]

Требуется: Напишите функцию, которая возвращает массив, являющийся суммой соответствующих элементов двух данных массивов.

Решение:

python

```
import numpy as np

arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
result = arr1 + arr2
print(result)  # Output: [5, 7, 9]
```

Теоретические вопросы:

Теоретический вопрос 1 (ИОПК-7.2): Опишите процесс работы с данными в библиотеке Pandas, начиная с загрузки данных и заканчивая их визуализацией. Включите описание основных функций и методов, которые могут быть использованы на каждом этапе.

Пример ответа:

Для работы с данными в Pandas необходимо:

Загрузка данных: Используются функции read_csv(), read_excel(), read_json() и др.

Предварительная обработка данных: Удаление пропусков с помощью dropna(), заполнение пропусков с помощью fillna(), удаление дубликатов с помощью drop_duplicates().

Анализ данных: Используются методы фильтрации query(), сортировки sort_values(), группировки groupby() и агрегации agg().

Визуализация данных: Использование функций plot(), hist(), boxplot() из Pandas и библиотек Matplotlib и Seaborn для более сложных визуализаций.

Теоретический вопрос 2 (ИОПК-7.2): Объясните основные принципы объектно-ориентированного программирования (ООП) в Python. Включите в ответ понятия классов, объектов, атрибутов и методов, а также наследования и полиморфизма.

Пример ответа:

ООП – это парадигма программирования, основанная на концепции объектов и классов.

Классы: Шаблоны для создания объектов. Определяются с помощью ключевого слова class.

Объекты: Экземпляры классов. Создаются путем вызова класса как функции.

Атрибуты: Переменные, принадлежащие классу или объекту.

Методы: Функции, принадлежащие классу или объекту.

Наследование: Позволяет создавать новые классы на основе существующих.

Полиморфизм: Возможность обработки разных типов данных через единый интерфейс.

Кейс (ИОПК-1.3, ИОПК-2.1, ИОПК-7.2):

Описание кейса: Анализ данных о продажах компании за последний год. Необходимо выполнить загрузку данных, их очистку, анализ и визуализацию результатов.

Формальная постановка задачи:

Загрузите данные о продажах из CSV-файла.
Очистите данные от пропусков и выбросов.
Выполните анализ продаж по месяцам и категориям продуктов.
Постройте визуализацию продаж за каждый месяц.
Решение и интерпретация:

Загрузка данных:
python

```
import pandas as pd
sales_data = pd.read_csv('sales_data.csv')
```

Очистка данных:
python

```
sales_data = sales_data.dropna()
sales_data = sales_data[sales_data['sales'] > 0]
```

Анализ данных:
python

```
monthly_sales = sales_data.groupby('month')['sales'].sum()
category_sales = sales_data.groupby('category')['sales'].sum()
```

Визуализация данных:
python

```
import matplotlib.pyplot as plt
monthly_sales.plot(kind='bar')
plt.show()
category_sales.plot(kind='pie', autopct='%1.1f%%')
plt.show()
```

Ответ должен содержать формальную постановку задач, их решение и интерпретацию полученных выводов.

Ключи к тестам и задачам:
Тест:
a) isinstance()
a) Определить функцию внутри другой функции
a) array(), arange(), linspace()
d) loc[]
a) requests
Задачи:

Решение задачи 1:
python

```
def increase_prices(prices):
    return [price * 1.1 for price in prices]
prices = [250000, 300000, 350000, 400000, 450000]
new_prices = increase_prices(prices)
print(new_prices)  # Output: [275000.0, 330000.0, 385000.0, 440000.0, 495000.0]
```

Решение задачи 2:

```
python

import numpy as np
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
result = arr1 + arr2
print(result)  # Output: [5, 7, 9]
```

Информация о разработчиках

Салаева А.С., методист Skillfactory