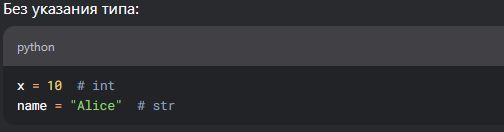
Лабораторная работа 1. Ответы на контрольные вопросы

1. Чем отличается интерпретируемый язык от компилируемого?

Интерпретируемый язык (Python) выполняет код построчно без предварительной компиляции, что упрощает отладку, но может работать медленнее. Компилируемый язык (C++, Java) преобразует весь код в машинный заранее, что ускоряет выполнение.

1. Примеры компилируемых языков: C++, Java, C#.
2. Примеры интерпретируемых языков - Python, JS, Ruby.
3. Как объявляются переменные в Python?



1. Как определить тип переменной? - Через функцию type()

6. Назначение print? - Выводить в терминале значение, которое получаем в результате выполнения кода

7. Функция для модуля числа - abs() из стандартной библиотеки



1. Недопустимые имена переменных:

* Ключевые слова (int)
* Дефис в названии
* Имя переменной начинается с цифры

1. Основные типы в python - int, float, str, bool, list, dict

Лабораторная работа 2.

1. Какой оператор предназначен для многократного повторения блока кода? - операторы while (для выполнения цикла, пока условие истинно) и for (для итерации по последовательностям (спискам, строкам, диапазонам)). Примеры:

for i in range(5): # Повторить 5 раз

print(i)

count = 0

while count < 5:

print(count)

count += 1

1. Перечислите операторы, применяемые в работе скрипта:

* If - для прокерки кратности;
* Цикл for - перебор элементов из списка;
* Оператор % - остаток от деления;
* Функция print() - вывод результата.

1. Параметры функции print и их значения:

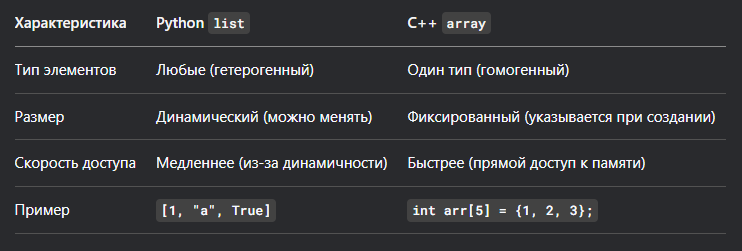
* print(1, 2, 3, sep="|") # Вывод: 1|2|3 - разделитель между аргументами;
* print("Hello", end="!") # Вывод: Hello! - символ в конце строки;
* print("Save to file", file=f) - поток вывода (пример - файл вместо консоли);

1. Список (list )— это изменяемая (мутабельная) упорядоченная коллекция элементов произвольных типов.

my\_list = [10, "text", True, [1, 2]] # Может содержать числа, строки, другие списки

1. Механизм обращения к элементам списка:

* Положительные индексы — начинаются с 0 (первый элемент); print(my\_list[0]) # 10
* Отрицательные индексы — отсчёт с конца (-1 - последний элемент); print(my\_list[-1]) # [1, 2]
* Срезы - получение подсписка; print(my\_list[1:3]) # ["text", True]

1. 

Лабораторная работа 3.

1. Общее у строки и списка в Python? - Строки и списки в Python являются последовательностями (sequence types). Они поддерживают индексацию, срезы (slicing), итерацию с помощью цикла for,  также методы для работы с элементами (например,len(), in). Однако строки неизменяемы (immutable), а списки изменяемы (mutable).
2. Функции класса str в Python:

* Split() - разбивка строки по разделителю;
* Join() - объединяет элементы списка в строку;
* Strip() - удаляет пробелы и спецсимволы с начала и конца строки;
* Upper(), lower() - преобразование регистра;
* Find(), index() -поиски подстроки;
* Replace() - замена подстроки;

1. Dict - это неупорядоченная коллекция пар «ключ-значение». Ключи должны быть уникальными и неизменяемыми (числа, строки, кортежи). Значения могут быть любого типа.
2. Функции dict:

* Keys() - возвращает список ключей;
* Values() - возвращает список значений;
* items() -возвращает пары «ключ-значение»;
* get() -безопасное получение значения по ключу;
* update() - объединяет словари.

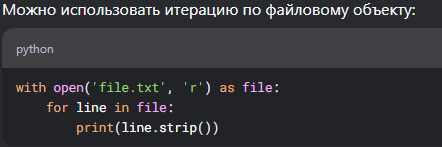
1. Чтение данных из текстового файла - Используется функция  open() с режимами 'r' (чтение), 'w' (запись), 'a' (дозапись). Для чтения всего файла — метод read(), построчно — readline() или readlines().
2. Различие функций чтения:

read() - читает весь файл в одну строку;

readline() - читает одну строку;

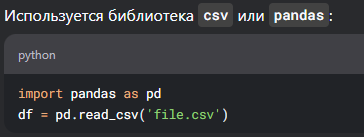
readlines() -  возвращает список строк.

1. Чтение многострочного файла без read():

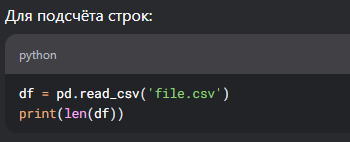


Лабораторная работа 4.

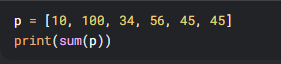
1. Структура CSV-файла. CSV (Comma-Separated Values) — текстовый формат, где данные разделены запятыми (или другими разделителями). Каждая строка представляет запись, а столбцы — поля.
2. Чтение CSV-файла в Python:



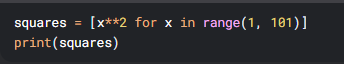
1. Пример обработки CSV:



1. Сумма элементов списка:

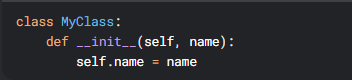


1. Генерации списка квадратов:



Лабораторная работа 5.

1. Класс - это шаблон для создания объектов, объединяющий данные (атрибуты) и методы (функции).
2. Метод \_\_init\_\_ - Это конструктор класса, вызывается при создании объекта. Инициализирует атрибуты:



1. Статические методы принадлежат классу, а не экземпляру. часто реализуются с использованием библиотек, таких как NumPy, SciPy и Pandas. Эти библиотеки предоставляют функции для вычисления статистических показателей, анализа данных и визуализации результатов. Создаются с декоратором @staticmethod и не требуют параметра self.
2. Параметр self - Ссылка на экземпляр класса, через которую можно обращаться к его атрибутам и методам.
3. Количество \_\_init\_\_ - в классе допускается только один \_\_init\_\_. Если определить несколько, последний переопределит предыдущие.

Лабораторная работа 6.

1. Функции трансформации изображений:

* rotate() - поворот;
* crop() - обрезка;
* resize() - изменение размера;
* transpose() - отражение.

1. Метод convert() - изменяет цветовой режим изображения (например, 'RGB', 'L' для grayscale).
2. Фильтрация изображений применяется для шумоподавления, выделения границ.



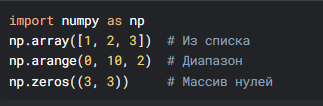
1. 4. Формат RGB Представляет цвета как комбинацию красного (Red), зелёного (Green) и синего (Blue) каналов.

5. Термины:

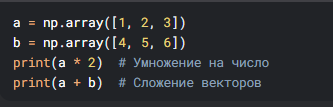
* Растр - изображение, представленное пикселями;
* Палитра - набор цветов, используемых в изображении;
* Глубина цвета - количество бит на пиксель (например, 8-bit для grayscale).

Лабораторная работа 7.

1. Отличие списка от массива NumPy - Массивы NumPy поддерживают векторные операции, имеют фиксированный тип и высокую производительность. Списки Python гибкие, но медленнее.
2. Создание массивов NumPy:



1. Векторные операции:



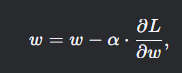
1. Генерация случайных чисел - np.random.rand(5) # 5 случайных чисел от 0 до
2. Типы диаграмм в matplotlib:

* plot() - линейный график;
* scatter() - точечная диаграмма;
* bar() - столбчатая диаграмма;
* hist() - гистограмма.

Лабораторная работа 8.

1. 1. Обучение алгоритмов - Алгоритмы обучаются на данных, минимизируя функцию потерь (например, MSE для регрессии). Процесс включает подбор параметров модели.

2. Линейная регрессия - Модель вида y=w1x1+w2x2+...+by=w1​x1​+w2​x2​+...+b, где w — веса, b — смещение.

3. Градиентный спуск - Итеративная оптимизация весов через вычисление градиента функции потерь и обновление параметровгде а - скокрость обучения.

4. Темп обучения - гиперпараметр, определяющий шаг градиентного спуска. Слишком большой αα может привести к расходимости, слишком маленький — к медленному обучению.

5. Пример визуализации:

Для регрессии можно построить график исходных точек и предсказаний модели:

1. 