### Вопросы по Python

#### 1. Типы данных, операторы, переменные, константы

- \*\*Типы данных\*\*: числа (int, float), строки (str), логические (bool), списки (list), кортежи (tuple), множества (set), словари (dict).

- \*\*Операторы\*\*: арифметические (+, -, \*, /, //, %, \*\*), логические (and, or, not), сравнения (==, !=, >, <, >=, <=), присваивания (=, +=, -=, \*=, /=, //=, %=, \*\*=).

- \*\*Переменные\*\*: именованные объекты, которые хранят значения. Пример: `x = 5`.

- \*\*Константы\*\*: переменные, значения которых не изменяются. В Python обычно обозначаются заглавными буквами, например, `PI = 3.14159`.

#### 2. Операторы ветвления, выбора, цикла

- \*\*Ветвления\*\*: `if`, `elif`, `else`.

- \*\*Циклы\*\*: `for`, `while`.

- \*\*Пример\*\*:

```python

if x > 0:

print("Positive")

elif x < 0:

print("Negative")

else:

print("Zero")

for i in range(5):

print(i)

while x > 0:

print(x)

x -= 1

```

#### 3. Массивы одномерные и многомерные

- В Python массивы обычно реализуются с помощью списков.

- \*\*Одномерный массив\*\*:

```python

arr = [1, 2, 3, 4, 5]

```

- \*\*Многомерный массив\*\*:

```python

arr = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

```

#### 4. Строки. Набор методов для работы со строками

- \*\*Строки\*\*: неизменяемые последовательности символов.

- \*\*Методы\*\*: `lower()`, `upper()`, `split()`, `join()`, `replace()`, `find()`, `format()`, `strip()`.

- \*\*Пример\*\*:

```python

s = "Hello, World!"

print(s.lower()) # "hello, world!"

print(s.upper()) # "HELLO, WORLD!"

print(s.split()) # ['Hello,', 'World!']

```

#### 5. Функции (методы)

- \*\*Функции\*\*: блоки кода, которые выполняют определенные задачи. Определяются с помощью `def`.

- \*\*Пример\*\*:

```python

def add(a, b):

return a + b

result = add(2, 3)

print(result) # 5

```

#### 6. Файловый ввод-вывод. Работа с файлами

- \*\*Открытие файла\*\*: `open()`, режимы (`r`, `w`, `a`, `b`).

- \*\*Чтение\*\*: `read()`, `readline()`, `readlines()`.

- \*\*Запись\*\*: `write()`, `writelines()`.

- \*\*Закрытие\*\*: `close()`.

- \*\*Пример\*\*:

```python

with open('example.txt', 'w') as file:

file.write("Hello, World!")

```

#### 7. Класс как абстрактный тип. Создание и использование классов

- \*\*Класс\*\*: шаблон для создания объектов (экземпляров).

- \*\*Определение класса\*\*: с помощью `class`.

- \*\*Пример\*\*:

```python

class Person:

def \_\_init\_\_(self, name, age):

self.name = name

self.age = age

def greet(self):

print(f"Hello, my name is {self.name}.")

p = Person("Alice", 30)

p.greet() # Hello, my name is Alice.

```

#### 8. Использование объектов и классов

- \*\*Объекты\*\*: экземпляры классов.

- \*\*Использование\*\*: создание экземпляров, вызов методов.

- \*\*Пример\*\*:

```python

p = Person("Bob", 25)

p.greet() # Hello, my name is Bob.

```

#### 9. Списки, кортежи, словари

- \*\*Списки\*\*: изменяемые последовательности элементов. Пример: `[1, 2, 3]`.

- \*\*Кортежи\*\*: неизменяемые последовательности элементов. Пример: `(1, 2, 3)`.

- \*\*Словари\*\*: коллекции пар "ключ-значение". Пример: `{'key1': 'value1', 'key2': 'value2'}`.

#### 10. Компоненты для работы с текстом

- \*\*Методы строк\*\*: `split()`, `join()`, `replace()`, `find()`, `format()`.

- \*\*Регулярные выражения\*\*: модуль `re` для поиска и замены.

- \*\*Пример\*\*:

```python

import re

text = "The rain in Spain"

result = re.findall(r"\bS\w+", text)

print(result) # ['Spain']

```

#### 11. Модули. Определение и подключение модулей

- \*\*Модули\*\*: файлы с расширением `.py`, содержащие функции и переменные.

- \*\*Импорт\*\*: с помощью `import` или `from ... import ...`.

- \*\*Пример\*\*:

```python

import math

print(math.sqrt(16)) # 4.0

```

#### 12. Как можно принять результат ввода на клавиатуре?

- \*\*Функция\*\*: `input()`.

- \*\*Пример\*\*:

```python

name = input("Enter your name: ")

print(f"Hello, {name}!")

```

#### 13. Как проверить, что все символы строки относятся к алфавитно-цифровым?

- \*\*Метод\*\*: `isalnum()`.

- \*\*Пример\*\*:

```python

s = "Hello123"

print(s.isalnum()) # True

```

#### 14. В чем разница между списком и кортежем?

- \*\*Список\*\*: изменяемая коллекция.

- \*\*Кортеж\*\*: неизменяемая коллекция.

- \*\*Пример\*\*:

```python

list\_example = [1, 2, 3]

tuple\_example = (1, 2, 3)

```

#### 15. Объясните функцию range

- \*\*Функция\*\*: генерирует последовательность чисел.

- \*\*Пример\*\*:

```python

for i in range(5):

print(i) # 0, 1, 2, 3, 4

```

#### 16. Что такое локальные и глобальные переменные в Python?

- \*\*Локальные переменные\*\*: определены внутри функции и недоступны вне её.

- \*\*Глобальные переменные\*\*: определены вне всех функций и доступны везде.

- \*\*Пример\*\*:

```python

x = "global"

def func():

x = "local"

print(x)

func() # local

print(x) # global

```

#### 17. В чем разница между func и func()?

- \*\*func\*\*: ссылка на функцию.

- \*\*func()\*\*: вызов функции.

#### 18. Объясните, как работает функция map

- \*\*Функция\*\*: применяет заданную функцию к каждому элементу итерируемого объекта.

- \*\*Пример\*\*:

```python

def square(x):

return x \* x

result = map(square, [1, 2, 3, 4])

print(list(result)) # [1, 4, 9, 16]

```

#### 19. Переменные в Python передаются по ссылке или по значению?

- \*\*Объекты\*\*: передаются по ссылке.

#### 20. Как развернуть список?

- \*\*Метод\*\*: `[::-1]` или `reverse()`.

- \*\*Пример\*\*:

```python

lst = [1, 2, 3]

print(lst[::-1]) # [3, 2, 1]

```

#### 21. Что означает self в классе?

- \*\*self\*\*: ссылка на текущий экземпляр класса.

#### 22. Как объединить списки в Python?

- \*\*Операторы\*\*: `+` или метод `extend()`.

- \*\*Пример\*\*:

```python

lst1 = [1, 2, 3]

lst2 = [4, 5]

print(lst1 + lst2) # [1, 2, 3, 4, 5]

```

#### 23. В чем разница между списками и массивами?

- \*\*Списки\*\*: динамические, могут содержать элементы разных типов.

- \*\*Массивы\*\*: стат

ические, оптимизированы для числовых данных (модуль `array`).

#### 24. Назовите изменяемые и неизменяемые объекты

- \*\*Изменяемые\*\*: списки, множества, словари.

- \*\*Неизменяемые\*\*: кортежи, строки, числа.

#### 25. Как округлить число до трех десятичных знаков?

- \*\*Функция\*\*: `round()`.

- \*\*Пример\*\*:

```python

num = 3.14159

print(round(num, 3)) # 3.142

```

#### 26. Где быстрее поиск: в словарях или списках?

- \*\*Словари\*\*: O(1) — быстрее.

- \*\*Списки\*\*: O(n).

#### 27. В чем разница между модулем и пакетом?

- \*\*Модуль\*\*: один файл с кодом.

- \*\*Пакет\*\*: каталог с модулями и файлом `\_\_init\_\_.py`.

#### 28. Как увеличить и уменьшить целое число в Python?

- \*\*Увеличение\*\*: `x += 1`.

- \*\*Уменьшение\*\*: `x -= 1`.

#### 29. Как проверить, существует ли значение в списке?

- \*\*Оператор\*\*: `in`.

- \*\*Пример\*\*:

```python

lst = [1, 2, 3]

print(2 in lst) # True

```

#### 30. В чем разница между append и extend?

- \*\*append\*\*: добавляет один элемент.

- \*\*extend\*\*: добавляет все элементы итерируемого объекта.

#### 31. В чем разница между pass, continue и break?

- \*\*pass\*\*: ничего не делает, используется как заполнитель.

- \*\*continue\*\*: пропускает текущую итерацию.

- \*\*break\*\*: выходит из цикла.

#### 32. Какие типы данных поддерживаются в Python? (5)

- \*\*Основные\*\*: int, float, bool, str, list.

#### 33. Как выполняется обработка исключений в Python?

- \*\*Конструкции\*\*: `try`, `except`, `finally`.

- \*\*Пример\*\*:

```python

try:

x = 1 / 0

except ZeroDivisionError:

print("Cannot divide by zero")

finally:

print("Execution complete")

```

#### 34. Tkinter. Создание окна приложения. Кнопки. Позиционирование.

- \*\*Пример\*\*:

```python

import tkinter as tk

root = tk.Tk()

root.title("My Application")

button = tk.Button(root, text="Click Me")

button.pack()

root.mainloop()

```

#### 35. Tkinter. Текстовая метка Label, Поле ввода Entry и другие элементы

- \*\*Пример\*\*:

```python

import tkinter as tk

root = tk.Tk()

label = tk.Label(root, text="Name")

label.pack()

entry = tk.Entry(root)

entry.pack()

root.mainloop()

```

#### 36. Работа с базой SQLite

- \*\*Модуль\*\*: `sqlite3`.

- \*\*Пример\*\*:

```python

import sqlite3

conn = sqlite3.connect('example.db')

cursor = conn.cursor()

cursor.execute('''CREATE TABLE IF NOT EXISTS users (id INTEGER PRIMARY KEY, name TEXT)''')

cursor.execute('''INSERT INTO users (name) VALUES ('Alice')''')

conn.commit()

for row in cursor.execute('SELECT \* FROM users'):

print(row)

conn.close()

```

#### 37. Python полностью поддерживает ООП?

- Да, Python поддерживает все концепции ООП: инкапсуляцию, наследование, полиморфизм.

#### 38. Что может быть ключом в словаре?

- \*\*Ключи\*\*: неизменяемые объекты (строки, числа, кортежи).

#### 39. Для чего используется функция `\_\_init\_\_`?

- \*\*Функция\*\*: инициализация экземпляров класса.

#### 40. Что такое синхронный код и асинхронный код?

- \*\*Синхронный код\*\*: выполняется последовательно.

- \*\*Асинхронный код\*\*: позволяет выполнять задачи параллельно, не блокируя основной поток.

#### 41. Что такое генератор и чем отличается от итератора?

- \*\*Генератор\*\*: функция, возвращающая итератор с помощью `yield`.

- \*\*Итератор\*\*: объект, поддерживающий функцию `\_\_iter\_\_` и метод `\_\_next\_\_`.

### Вопросы по Java

#### 42. Назначения языка Java. Средства разработки приложений Java.

- \*\*Назначение\*\*: кроссплатформенная разработка, веб-приложения, мобильные приложения.

- \*\*Средства\*\*: JDK, IntelliJ IDEA, Eclipse.

#### 43. Типы данных и операции над ними (язык Java).

- \*\*Типы данных\*\*: примитивные (int, float, boolean, char), ссылочные (классы, массивы).

- \*\*Операции\*\*: арифметические, логические, побитовые, присваивания.

#### 44. Объектно-ориентированное программирование (Java).

- \*\*Концепции\*\*: классы и объекты, наследование, полиморфизм, инкапсуляция, абстракция.

#### 45. Работа со строками (Java).

- \*\*Класс\*\*: `String`.

- \*\*Методы\*\*: `length()`, `charAt()`, `substring()`, `indexOf()`, `toUpperCase()`, `toLowerCase()`.

- \*\*Пример\*\*:

```java

String s = "Hello";

System.out.println(s.toUpperCase()); // HELLO

```

#### 46. Геттеры и сеттеры в Java.

- \*\*Геттеры\*\*: методы для получения значения полей.

- \*\*Сеттеры\*\*: методы для установки значения полей.

- \*\*Пример\*\*:

```java

public class Person {

private String name;

private int age;

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

}

```

### Вопросы по Android Studio

#### 47. Что такое Activity? Объясните, что такое Activity и какие состояния характерны для него. Жизненный цикл Activity.

- \*\*Activity\*\*: основная компонента приложения, представляющая один экран.

- \*\*Состояния\*\*: `onCreate()`, `onStart()`, `onResume()`, `onPause()`, `onStop()`, `onDestroy()`, `onRestart()`.

- \*\*Пример\*\*:

```java

public class MainActivity extends AppCompatActivity {

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

}

}

```

#### 48. Что такое AndroidManifest. Кратко объясните, что такое AndroidManifest и каковы его функции.

- \*\*AndroidManifest\*\*: файл конфигурации приложения, содержит информацию о компонентах (Activity, Service, Receiver), разрешениях, библиотеках.

#### 49. Что такое Intent? Кратко объясните, что такое Intent и каковы его функции.

- \*\*Intent\*\*: объект, используемый для межкомпонентного взаимодействия. Функции: запуск Activity, Service, отправка широковещательных сообщений.

#### 50. Диалоговые окна Android. Какие диалоговые окна могут использоваться в Android приложении?

- \*\*Типы\*\*: `AlertDialog`, `ProgressDialog`, `DatePickerDialog`, `TimePickerDialog`.

#### 51. Виды Layout. Ключевые отличия и свойства. Какие виды разметок экрана (layout) существуют в Android.

- \*\*Виды\*\*: `LinearLayout`, `RelativeLayout`, `ConstraintLayout`, `FrameLayout`, `GridLayout`.

#### 52. Исполняемые файлы на Android.

- \*\*Формат\*\*: APK (Android Package).

- \*\*Содержимое\*\*: байт-код, ресурсы, манифест.

#### 53. Что такое View? Приведите примеры ViewGroup.

- \*\*View\*\*: базовый класс для всех компонентов интерфейса.

- \*\*ViewGroup\*\*: контейнер для других View. Примеры: `LinearLayout`, `RelativeLayout`, `ConstraintLayout`.

#### 54. Структура проекта в Android Studio.

- \*\*Компоненты\*\*: манифест, Java-код, ресурсы (layout, drawable, values), библиотеки, настройки Gradle.

#### 55. Что такое SDK Manager и AVD Manager.

- \*\*SDK Manager\*\*: инструмент для управления SDK, обновлений, пакетов.

- \*\*AVD Manager\*\*: инструмент для создания и управления виртуальными устройствами Android.

#### 56. Создание простого меню.

- \*\*Пример\*\*:

```java

@Override

public boolean onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.menu\_main, menu);

return true;

}

```

### Вопросы по HTML

#### 57. Основные теги HTML. Приведите пример простой HTML-страницы.

- \*\*Основные теги\*\*: `<html>`, `<head>`, `<body>`, `<title>`, `<h1>`, `<p>`, `<a>`, `<img>`.

- \*\*Пример\*\*:

```html

<!DOCTYPE html>

<html>

<head>

<title>My First Page</title>

</head>

<body>

<h1>Hello, World!</h1>

<p>Welcome to my website.</p>

</body>

</html>

```

#### 58. Какие форматы изображений могут быть использованы на веб-странице.

- \*\*Форматы\*\*: JPEG, PNG, GIF, SVG, WebP.

#### 59. Какие методы передачи данных существуют в HTML.

- \*\*Методы\*\*: `GET`, `POST`.

#### 60. Как вставить комментарий в HTML.

- \*\*Синтаксис\*\*:

```html

<!-- This is a comment -->

```

#### 61. Пример формы для ввода данных.

- \*\*Пример\*\*:

```html

<form action="/submit" method="post">

<label for="name">Name:</label>

<input type="text" id="name" name="name">

<input type="submit" value="Submit">

</form>

```

### Вопросы по CSS

#### 62. Основные стили CSS.

- \*\*Стили\*\*: `color`, `background-color`, `font-size`, `margin`, `padding`, `border`.

#### 63. Как подключить CSS к HTML-документу.

- \*\*Способы\*\*: inline, internal, external.

- \*\*Пример\*\* (external):

```html

<link rel="stylesheet" href="styles.css">

```

#### 64. Как задать цвет фона для страницы.

- \*\*Свойство\*\*: `background-color`.

- \*\*Пример\*\*:

```css

body {

background-color: #f0f0f0;

}

```

#### 65. Как создать выпадающий список.

- \*\*Пример\*\*:

```html

<select>

<option value="1">Option 1</option>

<option value="2">Option 2</option>

</select>

```

#### 66. Как сделать текст жирным.

- \*\*Свойство\*\*: `font-weight`.

- \*\*Пример\*\*:

```css

p {

font-weight: bold;

}

```

#### 67. Как задать размер шрифта.

- \*\*Свойство\*\*: `font-size`.

- \*\*Пример\*\*:

```css

p {

font-size: 16px;

}

```

#### 68. Как сделать текст курсивом.

- \*\*Свойство\*\*: `font-style`.

- \*\*Пример\*\*:

```css

p {

font-style: italic;

}

```

#### 69. Как сделать отступы вокруг элемента.

- \*\*Свойства\*\*: `margin`, `padding`.

- \*\*Пример\*\*:

```css

p {

margin: 10px;

padding: 5px;

}

```

#### 70. Как задать рамку вокруг элемента.

- \*\*Свойство\*\*: `border`.

- \*\*Пример\*\*:

```css

p {

border: 1px solid black;

}

```

### Вопросы по JS

#### 71. Основные функции JavaScript.

- \*\*Функции\*\*: `alert()`, `console.log()`, `getElementById()`, `addEventListener()`.

- \*\*Пример\*\*:

```javascript

document.getElementById("myButton").addEventListener("click", function() {

alert("Button clicked!");

});

```

#### 72. Как объявить и использовать переменные.

- \*\*Объявление\*\*: `var`, `let`, `const`.

- \*\*Пример\*\*:

```javascript

let x = 5;

const y = 10;

var z = x + y;

console.log(z); // 15

```

#### 73. Типы данных JavaScript.

- \*\*Типы данных\*\*: `number`, `string`, `boolean`, `object`, `undefined`, `null`, `symbol`.

#### 74. Как вывести сообщение на экран.

- \*\*Функции\*\*: `alert()`, `console.log()`.

- \*\*Пример\*\*:

```javascript

alert("Hello, World!");

console.log("Hello, Console!");

```

#### 75. Основные операторы JavaScript.

- \*\*Операторы\*\*: арифметические (`+`, `-`, `\*`, `/`), логические (`&&`, `||`, `!`), сравнения (`==`, `===`, `!=`, `!==`, `>`, `<`).

#### 76. Оператор ветвления if-else.

- \*\*Пример\*\*:

```javascript

let x = 5;

if (x > 0) {

console.log("Positive");

} else {

console.log("Non-positive");

}

```

#### 77. Оператор switch-case.

- \*\*Пример\*\*:

```javascript

let day = 3;

switch (day) {

case 1:

console.log("Monday");

break;

case 2:

console.log("Tuesday");

break;

case 3:

console.log("Wednesday");

break;

default:

console.log("Unknown day");

}

```

#### 78. Циклы for и while.

- \*\*Пример\*\*:

```javascript

for (let i = 0; i < 5; i++) {

console.log(i);

}

let j = 0;

while (j < 5) {

console.log(j);

j++;

}

```

#### 79. Как сделать задержку выполнения кода.

- \*\*Функция\*\*: `setTimeout()`.

- \*\*Пример\*\*:

```javascript

setTimeout(function() {

console.log("Executed after 2 seconds");

}, 2000);

```

#### 80. Асинхронные операции в JavaScript.

- \*\*Методы\*\*: `setTimeout()`, `setInterval()`, `Promise`, `async/await`.

- \*\*Пример\*\*:

```javascript

async function fetchData() {

let response = await fetch('https://api.example.com/data');

let data = await response.json();

console.log(data);

}

fetchData();

```

Если у вас возникнут дополнительные вопросы или потребуется более детальное объяснение по какому-либо из пунктов, пожалуйста, дайте знать!