**NAME : - GAJERA PAL**

**Python Theory Questions & Answers**

**1. Printing on Screen**

**ANS**

**Introduction to print() Function**

* The print() function is used to display output on the screen.
* It can take multiple arguments separated by commas.

**Formatted Output**

* **Using f-strings:**
* name = "Alice"
* age = 25
* print(f"My name is {name} and I am {age} years old.")
* **Using format():**
* print("My name is {} and I am {} years old.".format(name, age))
* **Using Concatenation:**
* print("My name is " + name + " and I am " + str(age) + " years old.")

**2. Reading Data from Keyboard**

**ANS**

**Using input() Function**

* input() reads user input as a string.
* Example:
* name = input("Enter your name: ")
* print("Hello,", name)

**Converting User Input to Different Data Types**

* **Reading an integer:**
* age = int(input("Enter your age: "))
* **Reading a float:**
* price = float(input("Enter the price: "))
* **Reading multiple values:**
* name, age = input("Enter your name and age: ").split()
* age = int(age)

**3. Opening and Closing Files**

**ANS**

**Opening a File**

* open("filename", mode) is used to open a file.

**Common File Modes:**

| **Mode** | **Description** |
| --- | --- |
| 'r' | Read mode (default) |
| 'w' | Write mode (overwrites file) |
| 'a' | Append mode (adds data to existing file) |
| 'r+' | Read and write mode |
| 'w+' | Write and read mode (overwrites file) |

**Closing a File**

* Always close a file using close() after operations.
* file = open("example.txt", "w")
* file.write("Hello, world!")
* file.close()

**4. Reading and Writing Files**

**ANS**

**Reading from a File**

* **Using read():** Reads the entire file.
* file = open("example.txt", "r")
* content = file.read()
* print(content)
* file.close()
* **Using readline():** Reads one line at a time.
* line = file.readline()
* **Using readlines():** Reads all lines as a list.
* lines = file.readlines()

**Writing to a File**

* **Using write():**
* file = open("example.txt", "w")
* file.write("This is a test.")
* file.close()
* **Using writelines():**
* file.writelines(["Line 1\n", "Line 2\n"])

**5. Exception Handling**

**ANS**

**Handling Errors with try-except-finally**

try:

num = int(input("Enter a number: "))

print(10 / num)

except ZeroDivisionError:

print("Cannot divide by zero!")

except ValueError:

print("Invalid input! Please enter a number.")

finally:

print("Execution completed.")

**Custom Exceptions**

* Custom exceptions can be created using classes:
* class CustomError(Exception):
* pass

**6. Class and Object (OOP Concepts)**

**ANS**

**Understanding Classes and Objects**

* A **class** is a blueprint for objects.
* An **object** is an instance of a class.
* class Person:
* def \_\_init\_\_(self, name, age):
* self.name = name
* self.age = age
* p = Person("Alice", 25)
* print(p.name, p.age)

**Local vs. Global Variables**

* **Local Variable:** Defined inside a function and accessible only within that function.
* **Global Variable:** Defined outside all functions and accessible throughout the program.
* x = 10 # Global variable
* def func():
* y = 5 # Local variable
* print(y)
* func()
* print(x)

**7. Inheritance**

**ANS**

**Types of Inheritance in Python**

| **Type** | **Description** |
| --- | --- |
| **Single** | One parent, one child |
| **Multilevel** | Child inherits from another child class |
| **Multiple** | One child inherits from multiple parents |
| **Hierarchical** | Multiple children inherit from the same parent |
| **Hybrid** | Combination of different types |

**Example: Single Inheritance**

class Parent:

def display(self):

print("This is the parent class.")

class Child(Parent):

pass

c = Child()

c.display()

**Using super() to Call Parent Class Methods**

class Parent:

def show(self):

print("Parent class")

class Child(Parent):

def show(self):

super().show()

print("Child class")

c = Child()

c.show()

**8. Method Overloading and Overriding**

**ANS**

**Method Overloading (Using Default Arguments)**

class Math:

def add(self, a, b, c=0):

return a + b + c

obj = Math()

print(obj.add(2, 3))

print(obj.add(2, 3, 4))

**Method Overriding (Redefining Parent Method in Child Class)**

class Parent:

def show(self):

print("Parent class method")

class Child(Parent):

def show(self):

print("Child class method")

obj = Child()

obj.show()

**9. SQLite3 and PyMySQL (Database Connectors)**

**ANS**

**Connecting to SQLite3 Database**

import sqlite3

conn = sqlite3.connect("mydatabase.db")

cursor = conn.cursor()

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

conn.commit()

conn.close()

**Connecting to MySQL using PyMySQL**

import pymysql

conn = pymysql.connect(host="localhost", user="root", password="", database="mydb")

cursor = conn.cursor()

cursor.execute("SELECT \* FROM users")

print(cursor.fetchall())

conn.close()

**10. Search and Match Functions (re Module in Python)**

**ANS**

**Using re.search() (Finds First Occurrence in String)**

import re

text = "Hello, my name is Alice."

match = re.search("name", text)

if match:

print("Found:", match.group())

**Using re.match() (Matches Only at Start of String)**

match = re.match("Hello", text)

if match:

print("Match found at the beginning!")