**Task No 2**

**Question no 1**

**Difference between Class and Object in Python. And give example.**

**Class:**

The **abstract blueprint** or **idealized prototype** that defines the potential and structure for what objects of that type should be. It’s a generalized description that encapsulates shared attributes and behaviors but doesn’t exist as a standalone entity.

**Example:**

**class Dog:**

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

**self.name = name**

**self.age = age**

**self.breed = breed**

**def bark(self):**

**print("Woof!")**

**Object:**

The **concrete realization** or **actual instance** of a class. It represents a specific embodiment of the class's blueprint with actual values for its attributes and the ability to perform actions defined by the class.

**Example:**

**my\_dog = Dog("Max", 3, "Golden Retriever")**

**your\_dog = Dog("Buddy", 5, "Labrador")**

**print(my\_dog.name)**

**my\_dog.bark()**

**print(your\_dog.breed)**

**your\_dog.bark()**

**Question no 2**

**Difference between Constructor Method (\_\_init\_\_) vs \_\_str\_\_ () Function:**

**\_\_init\_\_ (Constructor Method):**

**- Initializes an object when it's created.**

**- Sets the initial state of an object by assigning values to its attributes.**

**- Called automatically when an object is instantiated.**

**Example:**

**Class person:**

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

**self.name = name**

**self.age = age**

**def \_\_str\_\_ (self):**

**person = Person(name="Alice", age=30)**

**print(person)**

**\_\_str\_\_ (String Representation Method):**

**- Returns a string representation of an object**

**- Called when you want to print or display an object as a string**

**- Should return a human-readable string that represents the object**

**Example:**

**Class person:**

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

**self.name = name**

**self.age = age**

**def \_\_str\_\_ (self):**

**person = Person(name="Alice", age=30)**

**print(person)**

**Thank You**