09-09-2024 Python
Introduction to the Object Oriented Programming

Initializing Classes and Objects

Example of Using Constructors and Creating Classes

We use the self entry point (can be any variable but it is usually 'self') in order to initialize the constructor of the class. The entry point assists us in accessing the local properties of the class.

```
class Student:
    def __init__(self):
        self.a = 10
        print(id(self))

    def attendance():
        print("From attendance");

c = Student()
print(c.a)
```

Output

> *10*

What are Refernce Variable to Objects

From this example we can understand that when we create a new Object, the variable that we use to store the Object is not the actual object and it stores the memory reference to the Object.

```
class Student:
    def __init__(self):
        self.a = 10
        print(id(self))

    def attendance():
        print("From attendance");

s = Student()
print(s.a)
print(id(s))

s1 = s
print(id(s))
```

Output

> *1316912151376*
> *10*

> *1316912151376*
> *1316912151376*

Constructors with Parameters

```
class Student:
    def __init__(self,name,city):
        self.a = 10
        self.name = name
        self.city = city

    def m1(self):
        print(self.name)
        print(self.city)

s = Student("Oswald","Surat")
s.m1()
```

Output

> *Oswald*
> *Surat*

Using Local Variables inside a constructor

```
class Student:
    x = 'abc'
    y = 'xyz'
    def __init__(self):
        self.a = 10
        self.name = self.x
        self.city = self.y

    def m1(self):
        print(self.name)
        print(self.city)

s = Student()
s.m1()
```

Output

> *abc*
> *xyz*

Using Static Variables in Class

Here, we use the class itself to access the properties of it rather than the name of the object. These are called as static variables as we are accessing them without the use of an Object.

```
class Student:
    x = 'abc'
    y = 'xyz'

    def m1(self):
        print(Student.x)
        print(Student.y)

s = Student()
s.m1()
```

Output

> *abc*
> *xyz*

Passing Objects as Parameters in Objects

```
class Student:
    x = 'abc'
    y = 'xyz'

    def m1(self):
        print(self.x)
        print(self.y)
        print(id(self))

    def m2(self,student):
        print(id(self))
        student.m1()

s = Student()
s.m1()
s.m2(s)
print(id(s))
```

Output

> *abc*
> *xyz*
> *1551915072336*
> *1551915072336*
> *abc*
> *xyz*
> *1551915072336*
> *1551915072336*

Passing Objects as Parameters in Objects || Manipulating reference variables of Objects

```
class Student:
    x = 'abc'
    y = 'xyz'

    def m1(self):
        print(self.x)
        print(self.y)
        print(id(self))

    def m2(self,student):
        print(id(self))
        student.m1()
        print(student.x)
        print(self.x)

s = Student()
print(s.x)
s.m1()
s.x = 30
s.m2(s)
print(id(s))
```

Output

> *abc*
> *abc*
> *xyz*
> *2079694437200*
> *2079694437200*
> *30*
> *xyz*
> *2079694437200*
> *30*
> *30*
> *2079694437200*