

Behaviours of Class

Class contains two things as Characteristics and Behaviours.
Behaviours of class means the methods of that class.

In python there are three types of methods as

- 1.Instance method
- 2.Class method
- 3.Static method

Instance Methods in Python

- Instance methods are the most common type of methods in Python classes.
- These are so called because they can access unique data of their instance.
- If we have two objects each created from a Demo class, then they each may have different properties.
- Instance methods must have self as a parameter, but we don't need to pass this in every time.
- Self is another Python special term.
- Inside any instance method, we can use self to access any data or methods that may reside in our class.
- We won't be able to access them without going through self.
- Finally, as instance methods are the most common, there's no decorator needed.
- Any method we create will automatically be created as an instance method, unless we tell Python otherwise.

Static Methods in Python

- Static methods are methods that are related to a class in some way, but don't need to access any class-specific data.
- We don't have to use self, and we don't even need to instantiate an instance, we can simply call our method.
- The @staticmethod decorator was used to tell Python that this method is a static method.
- Static methods are great for utility functions, which perform a task in isolation.
- They don't need to (and cannot) access class data.
- They should be completely self-contained, and only work with data passed in as arguments.

Class Methods in Python

- Class methods are the third and final OOP method type to know.
- Class methods know about their class.
- They can't access specific instance data, but they can call other static methods.
- Class methods don't need self as an argument, but they do need a parameter called cls. This stands for class, and like self, gets automatically passed in by Python.
- Class methods are created using the @classmethod decorator.

Consider below application which demonstrates types of Behaviours

```
print("---- Marvellous Infosystems by Piyush Khairnar-----")
```

```
print("Demonstration of Behaviours of Class")
```

```
class Demo:
```

```
    def __init__(self):
```

```
        self.i = 0
```

```
        self.j = 0
```

```
    def fun(self):
```

```
        print("Inside instance")
```

```
    @classmethod
```

```
    def gun(cls):
```

```
        print("Inside class method")
```

```
    @staticmethod
```

```
    def sun():
```

```
        print("Inside static")
```

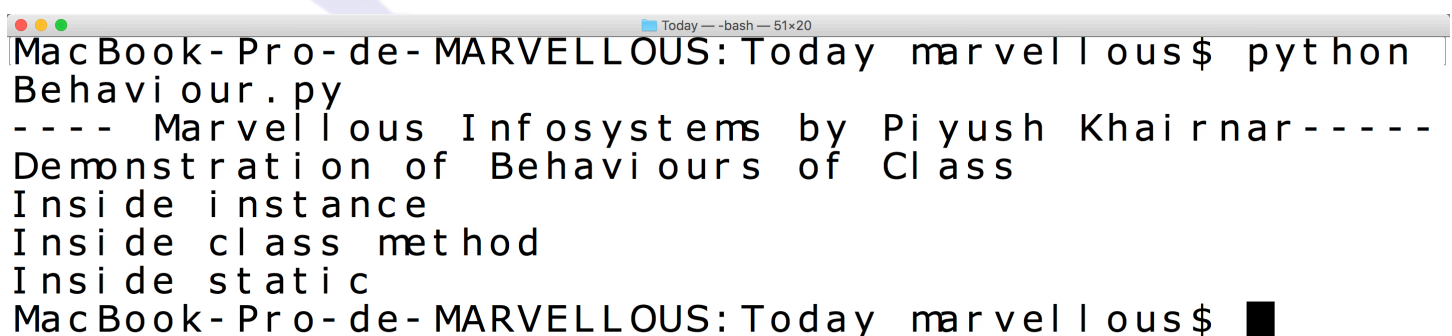
```
obj1 = Demo()
```

```
obj1.fun()
```

```
Demo.gun()
```

```
Demo.sun()
```

Output of above application



```
MacBook-Pro-de-MARVELLOUS:Today marvellous$ python
Behavi our . py
---- Marvellous Infosystems by Piyush Khairnar----
Demonstration of Behaviours of Class
Inside instance
Inside class method
Inside static
MacBook-Pro-de-MARVELLOUS:Today marvellous$
```