CLASS VS INSTANCE VARIABLE

Class Variable

* It is declared inside the class definition (but outside any of the instance methods).

* Class variables are those variables where you have only one copy of the variable which is shared with all the instance of the class.

* It is also called a static variable

In [1]:
```
class Student:
    prog_leader = 'Mrs. Amita Shukla' # class variable
    college = 'Amity University Lucknow'
    programme = 'MBA'
    def __init__(self, name,roll_no,age,specialization):
        self.name = name  # instance variable
        self.roll_no = roll_no
        self.Age = age
        self.specialization = specialization
```

In [2]:
```
John = Student("John",'A7001235986',21,'IT')
Harry = Student("Harry",'A7001920017',22,'Operations')
Radha = Student("Radha",'A7001920011',23,'HR')
```

Class variables can be accessed using either class name or object reference.

In [3]:
```
print(Student.college)
```

```
Amity University Lucknow
```

In [4]:
```
obj1 = Student("Python",'A7001235986',22,'Marketing')
```

In [5]:
```
print(obj1.college)
```

```
Amity University Lucknow
```

Instance variable are unique to all the instances of the class

- Id function return the address of the object

In [7]:
```
id(John.roll_no) == id(Harry.roll_no)
```

Out[7]:
```
False
```

In [8]:
```
id(Radha.name)==id(John.name)
```

Out[8]:
```
False
```

Class variables are common to all instances of a class

In [9]:
```
id(John.prog_leader) == id(Harry.prog_leader)
```

Out[9]:
```
True
```

In [12]:
```
print("Name = ",John.name, "\nRoll no. = ",John.roll_no,"\nProgramme = ", Student.programme ,"\nCollege name = ",Stuc
```

```
Name =  John
Roll no. =  A7001235986
Programme =  MBA
College name =  Amity University Lucknow
```

In [13]:
```
print(Student.programme)
```

```
MBA
```

In [14]:
```python
print(John.programme)
```

```
MBA
```

Modifying a class variable

In [15]:
```python
Harry.prog_leader
```

Out[15]:
```
'Mrs. Amita Shukla'
```

In [16]:
```python
Student.prog_leader = 'Mr. Satya Nadella'
```

In [17]:
```python
John.prog_leader
```

Out[17]:
```
'Mr. Satya Nadella'
```

Instance Variable

* Instance Variable declared inside the constructor method of class (the **init** method).

* Every instance of that class (object) has it's own copy of that variable.

Modifying a class variable

In [20]:
```python
class Employee:
    Holiday = 13
    pass

Rohan= Employee()
Rohit = Employee()
```

```
Rohan.fname="Rohan"
Rohan.lname ="Singh"
Rohan.salary=45000
Rohan.profile="Data Scientist"
Rohan.Experience = '6 Yrs'

Rohit.fname="Rohit"
Rohit.lname ="Singh"
Rohit.salary=56000
Rohit.profile="Data Analyst"
Rohit.Experience = '8 Yrs'
```

In [21]:
```
print(Rohit.Holiday)
```

```
13
```

In [22]:
```
print(Employee.Holiday)
```

```
13
```

In [23]:
```
print(Rohan.__dict__)
```

```
{'fname': 'Rohan', 'lname': 'Singh', 'salary': 45000, 'profile': 'Data Scientist', 'Experience': '6 Yrs'}
```

In [24]:
```
print(Employee.__dict__)
```

```
{'__module__': '__main__', 'Holiday': 13, '__dict__': <attribute '__dict__' of 'Employee' objects>, '__weakref__': <a
ttribute '__weakref__' of 'Employee' objects>, '__doc__': None}
```

Here Python Interpreter creates a new instance variable in Rohan

In [25]:
```
Rohan.Holiday
```

Out[25]:
```
13
```

In [26]:

```
Rohan.Holiday=14
```

In [27]:

```
print(Rohan.__dict__)
```

```
{'fname': 'Rohan', 'lname': 'Singh', 'salary': 45000, 'profile': 'Data Scientist', 'Experience': '6 Yrs', 'Holiday':
14}
```

In [28]:

```
print(Employee.__dict__)
```

```
{'__module__': '__main__', 'Holiday': 13, '__dict__': <attribute '__dict__' of 'Employee' objects>, '__weakref__': <a
ttribute '__weakref__' of 'Employee' objects>, '__doc__': None}
```

In [29]:

```
Employee.Holiday =15
```

In [30]:

```
print(Employee.__dict__)
```

```
{'__module__': '__main__', 'Holiday': 15, '__dict__': <attribute '__dict__' of 'Employee' objects>, '__weakref__': <a
ttribute '__weakref__' of 'Employee' objects>, '__doc__': None}
```

In [ ]:

```

```