ACCESSING METHODS OF CLASS:

```
I/P - class HelloWorld:
        '''This is my HelloWorld class'''

            def display(self):
                Print("Hello World")

            h = HelloWorld()

            h.display()
```

O/P - Hello World.

CONSTRUCTOR:

```
I/P - class student:

        def __init__(self):

            self.name = 'XYZ'

            self.rno = 2

            self.marks = 100

    s1=student()

    Print(s1.name, s1.rno, s1.marks)
```

O/P - XYZ 2 100

I/P - class Test:

```
    def __init__(self):
        Print("constructor is executed")

    def fun(self):
        Print('Function is executed")

t1 = Test()
t2 = Test()

t1.fun()
t2.fun()
```

t1, t2 are objects

O/P - constructor is executed
constructor is executed
Function is executed
Function is executed

OOP :→ object
↳ class
↳ variables
↳ Methods
↳ Constructors
↳ __init__()
↳ "self" is the first parameter

↳ constructor

↳ It runs (or) executes automatically without (self) constructor

↓

It runs when object is created.

I/p - class bike:

```
def __init__(self, c, m, e, h, s):
    self.color = c
    self.mileage = m
    self.engno = e
    self.horsepower = h
    self.speed = s

def updatemaxspeed(self, s):
    self.speed = s

def updatemaxhorsepower(self, h):
    self.horsepower = h

activa = bike("Blue", "45", "ABC123", "8.29", "45")

Print(activa.speed)
```

O/p - 45

```
activa.updatemaxspeed(150)
Print(activa.speed)
```

O/p - 150

Creating a Bank Account:

I/p -

```
class bankaccount:
    def __init__(self,a,s):
        self.accno = a
        self.saving = s
        Print("Account Created")
    def deposite(self,d):
        Print("Deposite Successful")
        self.saving += d
    def withdraw(self,w):
        if(self.saving >= w):
            Print("Withdraw Successful")
            self.saving = -w
        else:
            Print("Insufficient funds")
    def display(self):
        Print("Saving balance = ", self.saving)
```

```
pavani = bankaccount("173919", 20000)
```

O/p - Account created

```
Pavani.display()
```

O/p - saving balance = 20000

```
Pavani.deposite(20000)
```

O/p - Deposite successful

```
Pavani.withdraw(10000)
```

O/p - Withdraw successful.

```
Pavani.display()
```

O/p - saving balance = 30000

* For Private one, we give syntax as self.__accno, etc

CONSTRUCTOR WITH MULTIPLE PARAMETERS:

I/P -

```
class student:
    def __init__(self, name, rno, marks):
        self.name = name
        self.rno = rno
        self.marks = marks
    def display(self):
        Print('Name:{}, RollNo:{}, Marks:{}'
              \format(self.name, self.rno, self.
                      marks))
s1 = student('PAVANI', 90, 100)
s1.display()
s2. student('SITA', 13, 90)
s2.display()
```

O/P - Name: Pavani, RollNo: 90, Marks: 100

Name: SITA, RollNO: 13, Marks: 90

INSTANCE VARIABLE:

I/p -

```
class student:
    def __init__(self, name, rno, marks):
        self.name = name
        self.rno = rno
        self.marks = marks

    def display(self):
        Print('Name :{}, Roll NO:{}, Marks:{}'.
            \format(self.name, self.rno, self.
                marks))

S1 = student('PAVANI', 90, 100)
Print(S1.__dict__)
Print(S1.name)
S1.display()
S2 = student('SITA', 13, 90)
S2.display()
```

O/p - {'name': 'PAVANI', 'rno': 90, 'marks': 100}

PAVANI

Name: PAVANI, Roll NO: 90, Marks: 100.

Name: SITA, Roll NO: 13, Marks: 90

STATIC VARIABLE:

I/p - Class student:

```
    College = "ST. PETER'S ENGINEERING COLLEGE"

    def __init__(self, name, rno, marks):

        self.name = name
        self.rno = rno
        self.marks = marks

    def display(self):

        Print('Name : {}, RollNo: {}, Marks: {}.
              \format(self.name, self.rno, self.
                                        marks)
                                        end=' ')

        Print('college: ', self.college)

S1 = student('PAVANI', 90, 100)

S1.display()

S2 = student ('SITA', 13, 90)

S2.display()
```

O/p - Name: 'PAVANI', RollNo: 90, Marks: 100,

College : ST. PETER'S ENGINEERING COLLEGE

Name = SITA, RollNo : 13 ; Marks : 90

College : ST. PETER'S ENGINEERING COLLEGE

CHANGING THE STATIC VARIABLE USING S1:

(101)

```
S1.college = "SRI GAYATRI JUNIOR COLLEGE"
S1.display()
S2.display()
```

o/p - Name: PAVANI, ROLL NO: 90, Marks: 100,
College: SRIGAYATRI JUNIOR COLLEGE

Name: SITA, ROLL NO: 13, Marks: 90
College: ST. PETER'S ENGINEERING COLLEGE

LOCAL VARIABLE:

```
i/p - class Test:
        def P1(self):
            x = 40
            Print(x)
        def P2(self):
            y = 25
            Print(y)
    t = Test()
    t.P1()
    t.P2()
```

o/p - 40
25