Access Modifiers OR Access Specifiers

Access modifiers define the accessibility of members of the class.
Python provides 3 access modifiers.

1. Private
2. Protected
3. Public

Private members of the class prefix with __ (double underscore)
Protected members of the class prefix with _ (single underscore)
Public members of the class are not prefix with any underscore

	Private	**Protected**	**Public**
Within class	YES	YES	YES
Within sub class (Inherited class)	NO	YES	YES
Outside the class	NO	NO	YES

Data hiding is achieved by declaring data members of the class private.

Example	**Output**

class Alpha:
  def __init__(self):
  self.__x=100 # private instance variable
  self.y=200 # public instance variable
 	
obj1=Alpha()
#print(obj1.__x)
print(obj1.y)
	200

Private data members or data is manipulated by using methods of same class. Private data is accessible outside the class using public methods.

class Employee: def __init__(self,e,en,s):	Output EmloyeeNo 101,

```
        self.__empno=e
        self.__ename=en
        self.__salary=s
    def print_employee(self):
        print(f'''EmloyeeNo {self.__empno},
EmployeeName {self.__ename},
Salary {self.__salary}''')



emp1=Employee(101,"naresh",50000)
emp1.print_employee()
emp2=Employee(102,"suresh",65000)
emp2.print_employee()
```

```
EmployeeName naresh,
Salary 50000
EmloyeeNo 102,
EmployeeName suresh,
Salary 65000
```

Instance methods defined inside class perform 2 operations.

1. Setter Operation
2. Getter Operation

An operation which modifies the values of the object is called setter operation.
An operation which does not modify the values of the object is called getter operation.

Example

```
class Date:
    def __init__(self,d,m,y):
        self.__day=d
        self.__month=m
        self.__year=y
    def getDay(self):
        return self.__day
    def getMonth(self):
        return self.__month
    def getYear(self):
        return self.__year
    def setYear(self,y):
        self.__year=y
    def setMonth(self,m):
```

```
        self.__month=m
    def setDay(self,d):
        self.__day=d


name="Naresh"
dob=Date(5,4,2000)
print(f'name :{name}')
print(f'{dob.getDay()}/{dob.getMonth()}/{dob.getYear()}')
dob.setYear(2001)
print(f'{dob.getDay()}/{dob.getMonth()}/{dob.getYear()}')
dob.setMonth(6)
dob.setDay(3)
print(f'{dob.getDay()}/{dob.getMonth()}/{dob.getYear()}')
```

Output

```
name :Naresh
5/4/2000
5/4/2001
3/6/2001
```

Example

```
# Write a program to find area of triangle

class Triangle:
    def __init__(self,b,h):
        self.__base=b
        self.__height=h
    def find_area(self):
        return 0.5*self.__base*self.__height

t1=Triangle(1.5,2.5)
area1=t1.find_area()
t2=Triangle(3.5,2.5)
area2=t2.find_area()

print(f'Area of triangle1 is {area1:.2f}')
print(f'Area of triangle2 is {area2:.2f}')
```

Output

Area of triangle1 is 1.88
Area of triangle2 is 4.38

Example

```
class Account:
    def __init__(self,a,cn,b):
        self.__accno=a
        self.__cname=cn
        self.__balance=b
    def print_account(self):
        print(f"'AccountNo :{self.__accno}
CustomerName :{self.__cname}
CustomerBalance :{self.__balance}"')
    def deposit(self,t):
        self.__balance+=t
    def withdraw(self,t):
        if self.__balance<t:
            print("Insuff Balance")
        else:
            self.__balance-=t

a1=Account(101,"naresh",56000)
a1.print_account()
a1.deposit(5000)
a1.print_account()
a1.withdraw(90000)
a1.withdraw(20000)
a1.print_account()

list1=list()
list1.append(10)
list1.append(20)
list1.remove(10)
print(list1)
```

Output

```
AccountNo :101
CustomerName :naresh
CustomerBalance :56000
AccountNo :101
```

```
CustomerName :naresh
CustomerBalance :61000
Insuff Balance
AccountNo :101
CustomerName :naresh
CustomerBalance :41000
[20]
```

Example

```
class A:
    def __init__(self):
        self.x=100
    def m1(self):
        print("instance method")


obj1=A()
obj1.m1()
print(obj1.x)
```

Output

```
instance method
100
```

Class variable