In [ ]:
```
#Agenda of the day:
                          1. Inheritance
                                - Single Level
                                - Multi Level
                                - Heiraicial
                                - Multiple
                          2. Polymorphsim
                                   - Complile Time  (method overloading)
                                   - RunTime    (method overriding)
                          3. Data Abstraction

                          4. Data Encapusualtion & Data Hiding
```

In [ ]:
```
#Inheritance:
     A class(derived or child class) which inherits or accuring the
     properties of another class (base or parent class) id called
        Inheritance.
                  inheritance
    Parent class--------->     Child class
        (to accuring the properties of parent to child)
#Note: By Using interitance concept we make code resuablity.
```

In [15]:
```
#Single_Level:
class arthematic:                    #parent-super-base-class
    a = 10
    b = 20
    def add(self):
        sums = self.a+self.b
        print("sum of a and b is",sums)
class addition(arthematic):                   #child-sub-derived-class
    c = 50
    d = 10
    def sub(self):
        subs =self.c-self.d
        print("subtraction of c and d is:",subs)
ob = addition()
print(ob.c)
print(ob.d)
ob.sub()
print(ob.a)
print(ob.b)
ob.add()
```

```
50
10
subtraction of c and d is: 40
10
20
sum of a and b is 30
```

In [ ]:
```
#Multi-Level Inheritance:
parent1class----->parent2class---->child2class

                  (child & parent)
```

In [53]:
```
#Example:(Multilevel---->one or more parent class)
class addition:
    c=50
    d=100                       #class variables
    def __init__(self,a,b):
        self.a=a
        self.b=b                        #a,b instance variables
    def add(self,a,b):
        sums = self.a+self.b
        print("sum of a and b is:",sums)
class substraction(addition):           #level1-inheritance
    def sub(self,a,b):
        subs = self.b-self.a
        print("substraction of b and a is:",subs)
class multiplication(substraction):             #level-2 inheritance
    def mul(self,a,b):
        multi = self.a * self.b
        print("multiplication of a and b is",multi)
a = int(input())
b = int(input())

obj= multiplication(a,b)
obj.add(a,b)
obj.mul(a,b)
obj.sub(a,b)
obj.c
```

```
10
20
sum of a and b is: 30
multiplication of a and b is 200
substraction of b and a is: 10
```

Out[53]:
```
50
```

In [ ]:
```
#Multiple Inheritance:
 A class which is inherits the properties of more than one parent class.
```

In [63]:
```
#Multile Inheritance: (2 Baseclasses, one child class)
class Father:                       #baseclass-1
    a =50
    b =100
    print(a,b)
    def parent1info(self):
        print("This is Parent1 class")

class Mother:                              #baseclass-2
    def parent2info(self):
        print("This is Parent2 class")
class Uncle:
    def uncleinfo(self):
        print("This is Uncle Class")

class kid(Father,Mother,Uncle):     #multiple inheritance
    def childinfo(self):
        print("This is child1 class")
class kid2(Father,Mother):
    def child2info(self):
        print("This is child2 class")
obj = kid()
obj1 = kid2()
obj.childinfo()
obj.parent2info()
obj.parent1info()
obj.uncleinfo()
obj1.parent1info()
```

```
50 100
This is child1 class
This is Parent2 class
This is Parent1 class
This is Uncle Class
This is Parent1 class
```

In [ ]:
```
#Hierarchical Inheritance:(1 Base Class, 2 Derived classes)
```

In [76]:
```
#Example: Hierarchical Inheritance
class Father:
    def Fatherinfo(self):
        print("This is Main Base Class")
class Son(Father):
    def child1info(self):
        print("This is Child1 Class")
class Daughter(Father):
    def child2info(self):
        print("This is Child2 Class")
objs = Son()
objs.child1info()
objs.Fatherinfo()
#objs.child2info()
objd = Daughter()
objd.child2info()
objd.Fatherinfo()
#objd.child1info()
objf=Father()
objf.Fatherinfo()
#objf.child2info()
```

```
This is Child1 Class
This is Main Base Class
This is Child2 Class
This is Main Base Class
This is Main Base Class
```

In [ ]: