In [ ]:
```
Concept Of Inheritance
Whatever  variables methods and constrcutors available to the parent class  by default availrabe
to the child class we are not required to write the code again and again
Basic use of Inheritance is Code Reusability
```

In [ ]:
```
Important Terminologies of Inheritance
Parent class/Base class/Super class --> Is that class which is being inherited
child class/derived class/Sub class --> is that class which is inheriting the properties
and behaviour from parent class
```

In [5]:
```
#Syntax of implement inheritance
class Parent:
    a=10
    def __init__(self):
        self.b=20

    def m1(self):
        print("Parent class instance method")

    @classmethod
    def m2(cls):
        print("Parent class class method")

    @staticmethod
    def m3():
        print("Parent class static method")

class Child(Parent):
    pass
c=Child()
print(c.a)
print(c.b)
c.m1()
c.m2()
c.m3()
```

```
10
20
Parent class instance method
Parent class class method
Parent class static method
```

In [ ]:
```
Types of Inheritance
Four type of inheritance
1.Single level Inheritance
2.Multi Level Inheritance
3.Hierarchical Inheritance
4.Multiple Interitence
```

In [ ]:
```
Single Level Inheritance --> the concept of inheriting properties and behaviour
from one class to another class is known as single  level inheritance(One parent and One child)

Example:
    Father----> You
```

In [6]:
```
class parent:
    def m1(self):
        print("Parent class m1 method")
class child(parent):
    def m2(self):
        print("Child class m2")
c=child()
c.m1()
c.m2()
```

```
Parent class m1 method
Child class m2
```

In [ ]:
```
Multilevel inheritance: If you want inheriting the properties of multiple class
    into a single class that type of inheritance is knownas multilevel inheritance

Example: GrandFather  ---> Father ----> You
```

In [10]:
```
class Tradiitonal_phone:
    def call(self):
        print("Calling Functionality")
    def sms(self):
        print("Sms functionality")

class Modern_Phone(Tradiitonal_phone):
    def camera(self):
        print("camera feature is there")

    def radio(self):
        print("Radio feature is there")

    def music(self):
        print("music feature is there")
    def video(self):
        print("Video feature is there")
class Iphone(Modern_Phone):
    def security(self):
        print("Secuirty feature is there")
    def siri(self):
        print("Siri is available")
x=Iphone()
x.call()
x.siri()
x.video()
```

```
Calling Functionality
Siri is available
Video feature is there
```

In [ ]:
```

```

In [ ]:
```
Hierarchical Inheritance -->The concept of inheriting the properties from one class to multiple
class which are present at the same level.
Example --> Traditional Phone ---> Smartphone
            Traditional Phone  ---> SmartWatch
```

In [15]:
```
class Tradiitonal_phone:
    def call(self):
        print("Calling Functionality")
    def sms(self):
        print("Sms functionality")

class Modern_Phone(Tradiitonal_phone):
    def camera(self):
        print("camera feature is there")

    def radio(self):
        print("Radio feature is there")

    def music(self):
        print("music feature is there")
    def video(self):
        print("Video feature is there")

class SmartWatch(Tradiitonal_phone):
    def calorie(self):
        print("WOW IT HAS CALORIE MESUREMENT FEATURE")
    def step(self):
        print("It also calcualte the steps of foot")
x=SmartWatch()
x.calorie()
x.step()
x.call()
```

```
WOW IT HAS CALORIE MESUREMENT FEATURE
It also calcualte the steps of foot
Calling Functionality
```

In [ ]:
```
Multiple Inheritance --> the concept of inheriting the properties and behaviour from multiple class
into one single class at a time is known as multiple inheritance.(Child class is one and
                                                                  parent class is Two)

Example  --> Mom --> SOn
             Dad --> Son
```

In [16]:
```
class Tradiitonal_phone:
    def call(self):
        print("Calling Functionality")
    def sms(self):
        print("Sms functionality")

class Modern_Phone:
    def camera(self):
        print("camera feature is there")

    def radio(self):
        print("Radio feature is there")

    def music(self):
        print("music feature is there")
    def video(self):
        print("Video feature is there")

class SmartWatch(Tradiitonal_phone,Modern_Phone):
    def calorie(self):
        print("WOW IT HAS CALORIE MESUREMENT FEATURE")
    def step(self):
        print("It also calcualte the steps of foot")
x=SmartWatch()
x.calorie()
x.step()
x.call()
x.music()
x.video()
```

```
WOW IT HAS CALORIE MESUREMENT FEATURE
It also calcualte the steps of foot
Calling Functionality
music feature is there
Video feature is there
```

In [20]:
```
class Tradiitonal_phone:
    def call(self):
        print("Calling Tradiitional phone Functionality")
    def sms(self):
        print("Sms functionality")

class Modern_Phone:
    def call(self):
        print("call modern phone feature is there")

    def radio(self):
        print("Radio feature is there")

    def music(self):
        print("music feature is there")
    def video(self):
        print("Video feature is there")

class SmartWatch(Tradiitonal_phone,Modern_Phone):
    def calorie(self):
        print("WOW IT HAS CALORIE MESUREMENT FEATURE")
    def step(self):
        print("It also calcualte the steps of foot")
x=SmartWatch()
x.calorie()
x.step()
x.call()
x.music()
x.video()
#Note: In python while giving the parent class name in child class it is necessary to take care
the order because methods are always called based on the order the Priority of method calling
is totally depend on parent class priority order.
```

```
WOW IT HAS CALORIE MESUREMENT FEATURE
It also calcualte the steps of foot
Calling Tradiitional phone Functionality
music feature is there
Video feature is there
```

In [ ]:
```
5.Cyclic Inheritance
It is not possible in python.
The concept of inheriting the properties and behaviour from one class class to itself into a cyclic
way. such type of inheritance are called cyclic inheritance.
```

In [ ]:
```
6.Hybrid Inheritance --> Combination of all the the types of inheritance
(single.multilevel,hiereachical,multiple)
```