In [1]:
```
# OOP: It is based on class and object.
```

In [2]:
```
#1.Class:It is a user defined data type. It consists two things. They a
re: i)Property(Attributes), and ii) Methods(Behaviors)
```

In [3]:
```
#2.Object:An object is a specific instances of a class.
```

In [4]:
```
#Example-01:
```

In [5]:
```
class Phone:
    def make_call(self):
        print("I am making a call.")
    def play_game(self):
        return 'I am palying a game.'
```

In [6]:
```
p1 = Phone()
```

In [7]:
```
p1.make_call()
```

```
I am making a call.
```

In [8]:
```
p1.play_game()
```

Out[8]:
```
'I am palying a game.'
```

In [9]:
```
#Example-02:
```

In [10]:
```
class Student:
    def result_pass(self):
        print("Congratulation! You pass the exam.")
    def result_fail(self):
        return"Sorry! You fail the exam."
```

In [11]:
```
r1 = Student()
```

In [12]:
```
r1.result_pass()
```

```
Congratulation! You pass the exam.
```

In [13]:
```
r1.result_fail()
```

Out[13]:
```
'Sorry! You fail the exam.'
```

In [14]:
```
#Example-03:
```

In [15]:
```
class Statement:
    def true(self):
        return'Your statement is true'
    def false(self):
        return'Your statement is false'
```

In [16]:
```
s1 = Statement()
```

In [17]:
```
s1.true()
```

Out[17]:
```
'Your statement is true'
```

In [18]:
```
s1.false()
```

Out[18]:
```
'Your statement is false'
```

In [19]:
```
#Example-04:
```

In [20]:
```
class Language:
    def best(self):
        print('You have the best command over English language.')
    def better(self):
        print('You have the better command over English language.')
    def good(self):
        return 'You the good command over English language.'
    def satisfactory(self):
        return 'You have the satisfactory command over English languag
e.'
    def poor(self):
        return'You have the poor command over English language.'
```

In [21]:
```
l1 = Language()
```

In [22]:
```
l1.best()
```

```
You have the best command over English language.
```

In [23]:
```
l1.better()
```

```
You have the better command over English language.
```

In [24]:
```
l1.good()
```

Out[24]:
```
'You the good command over English language.'
```

In [25]:
```
l1.satisfactory()
```

Out[25]:
```
'You have the satisfactory command over English language.'
```

In [26]:
```
l1.poor()
```

Out[26]:
```
'You have the poor command over English language.'
```

In [27]:
```
#a.Adding a parameter to a class:
```

In [28]:
```
#Example-01:
```

In [29]:
```
class Phone:
    def set_color(self,color):
        self.color=color
    def show_color(self):
        print(self.color)
    def set_cost(self,cost):
        self.cost=cost
    def show_cost(self):
        return self.cost
    def make_call(self):
        print('Make a call.')
    def play_game(self):
        return'Play a game'
```

In [30]:
```
p2 = Phone()
```

In [31]:
```
p2.set_color('Red')
```

In [32]:
```
p2.show_color()
```

```
Red
```

In [33]:
```
p2.set_cost(500)
```

In [34]:
```
p2.show_cost()
```

Out[34]:
```
500
```

In [35]:
```
p2.make_call()
```

```
Make a call.
```

In [36]:
```
p2.play_game()
```

Out[36]:
```
'Play a game'
```

In [37]:
```
# Example-02:
```

In [38]:
```
class Car:
    def set_color(self,color):
        self.color=color
    def show_color(self):
        return self.color
    def set_cost(self,cost):
        self.cost=cost
    def show_cost(self):
        return self.cost
    def set_hp(self,hp):
        self.hp = hp
    def show_hp(self):
        return self.hp
    def set_avg(self,avg):
        self.avg=avg
    def show_avg(self):
        print(self.avg)
    def drive(self):
        return'Drive a car'
    def buy(self):
        print("Buy a car")
```

In [39]:
```
c = Car()
```

In [40]:
```
c.set_color('Pink')
c.show_color()
```

Out[40]:
```
'Pink'
```

In [41]:
```
c.set_cost(50000000)
c.show_cost()
```

Out[41]:
```
50000000
```

In [42]:
```
c.set_hp(7000)
c.show_hp()
```

Out[42]:
```
7000
```

In [43]:
```
c.drive()
```

Out[43]:
```
'Drive a car'
```

In [44]:
```
c.buy()
```

```
Buy a car
```

In [45]:
```
#Example-03:
```

In [46]:
```
class Student:
    def set_name(self,name):
        self.name=name
    def show_name(self):
        return self.name
    def set_address(self,address):
        self.address=address
    def show_address(self):
        return self.address
    def set_parent(self,parent):
        self.parent = parent
    def show_parent(self):
        return self.parent
    def set_contact(self,contact):
        self.contact=contact
    def show_contact(self):
        print(self.contact)
    def set_age(self,age):
        self.age=age
    def show_age(self):
        return self.age
    def set_std(self,std):
        self.std=std
    def show_std(self):
        return self.std
    def set_bgroup(self,bgroup):
        self.bgroup = bgroup
    def show_bgroup(self):
        return self.bgroup
    def passed(self):
        return'Congratulation! Passed'
    def fail(self):
        print("Sorry! Failed")
```

In [47]:
```
s = Student()
```

In [48]:
```
s.set_name("Jigar")
s.show_name()
```

Out[48]:
```
'Jigar'
```

In [49]:
```
s.set_address("BalikaNagar")
s.show_address()
```

Out[49]:
```
'BalikaNagar'
```

In [50]:
```
s.set_parent("Anmol & Kavya")
s.show_parent()
```

Out[50]:
```
'Anmol & Kavya'
```

In [51]:
```
s.set_age(12)
s.show_age()
```

Out[51]:
```
12
```

In [52]:
```
s.set_std(7)
s.show_std()
```

Out[52]:
```
7
```

In [53]:
```
s.set_bgroup("B+")
s.show_bgroup()
```

```
Out[53]: 'B+'

In [54]: s.passed()

Out[54]: 'Congratulation! Passed'

In [55]: s.fail()

         Sorry! Failed

In [56]: s.set_contact(9856743210)

In [57]: s.show_contact()

         9856743210

In [58]: #A.Constructor:-

In [59]: #a.Creating constructor to add multiple parameters to a class:

In [60]: #Example-01:

In [61]: class Employee:
             def __init__ (self,name,age,gender,contact,address):
                 self.name=name
                 self.age=age
                 self.gender=gender
                 self.contact=contact
                 self.address=address
             def details(self):
                 print("The name of the employee is",self.name)
                 print("The age of the employee is",self.age)
                 print("The gender of the employee is",self.gender)
                 print("The contact of the employee is",self.contact)
                 print("The address of the employee is",self.address)
```

```
In [62]: c = Employee("Sonia",38,"Female",9876543210,'California')
```

```
In [63]: c.details()
```

```
The name of the employee is Sonia
The age of the employee is 38
The gender of the employee is Female
The contact of the employee is 9876543210
The address of the employee is California
```

```
In [64]: #Example-02:
```

```
In [65]: class Student:
             def __init__ (self,name,std,gender,parent,address,contact):
                 self.name=name
                 self.std=std
                 self.gender=gender
                 self.parent=parent
                 self.address=address
                 self.contact=contact
             def details(self):
                 print("The name of the student is",self.name)
                 print("The grade of the student is",self.std)
                 print("The gender of the student is",self.gender)
                 print("The parent of the student is",self.parent)
                 print("The contact of the employee is",self.contact)
                 print("The address of the employee is",self.address)
```

```
In [66]: s1 = Student("Bekham",9,'Male','Jason & Molly',"Texas",9808987665)
```

```
In [67]: s1.details()
```

```
The name of the student is Bekham
The grade of the student is 9
The gender of the student is Male
The parent of the student is Jason & Molly
The contact of the employee is 9808987665
The address of the employee is Texas
```

In [68]:
```
#Example-03:
```

In [69]:
```
class Car:
    def __init__ (self,name,color,cost,milage,hp,model):
        self.name = name
        self.color = color
        self.cost = cost
        self.milage = milage
        self.hp = hp
        self.model = model
    def details(self):
        print("The name of the car is",self.name)
        print("The color of the",self.name,"car is",self.color)
        print("The cost of the",self.name,"car is",self.cost)
        print("The milage of the",self.name,"car is",self.milage)
        print("The horse power of the",self.name,"car is",self.hp)
        print("The model of the",self.name,"car is",self.model)
```

In [70]:
```
c1 = Car("BMW","Red",70000000,250,450,"AM98XRG")
```

In [71]:
```
c1.details()
```

```
The name of the car is BMW
The color of the BMW car is Red
The cost of the BMW car is 70000000
The milage of the BMW car is 250
The horse power of the BMW car is 450
The model of the BMW car is AM98XRG
```

In [72]:
```
#b.Over-riding constructor(init method):
```

In [73]:
```
#Example-01: (Is also an example of Single Inheritance.)
```

In [74]:
```
class Employee:
    def __init__ (self,name,age,gender,contact,address):
            self.name=name
            self.age=age
            self.gender=gender
            self.contact=contact
            self.address=address
        def details(self):
            print("The name of the employee is",self.name)
            print("The age of the employee is",self.age)
            print("The gender of the employee is",self.gender)
            print("The contact of the employee is",self.contact)
            print("The address of the employee is",self.address)
```

In [75]:
```
e2 = Employee('Brad',34,"Male",9876543434,'California')
```

In [76]:
```
e2.details()
```

```
The name of the employee is Brad
The age of the employee is 34
The gender of the employee is Male
The contact of the employee is 9876543434
The address of the employee is California
```

In [77]:
```
class Job(Employee):
    def __init__ (self,name,age,gender,contact,address,height,weight):
        super(). __init__ (name,age,gender,contact,address)
        self.height = height
        self.weight = weight
    def job_details(self):
        print('The height of',self.name,'is',self.height,'ft.')
        print('The weight of',self.name,'is',self.weight,'kg.')
```

In [78]:
```
j1 = Job('Ronald',40,'Male',98765656439,'New York',6.1,85)
```

In [79]:
```
j1.details()
```

```
The name of the employee is Ronald
The age of the employee is 40
The gender of the employee is Male
The contact of the employee is 98765656439
The address of the employee is New York
```

In [80]:
```
j1.job_details()
```

```
The height of Ronald is 6.1 ft.
The weight of Ronald is 85 kg.
```

In [81]:
```
#xample-02:(Is also an example of Single Inheritance.)
```

In [82]:
```
class Student:
    def __init__ (self,name,std,gender,parent,address,contact):
        self.name=name
        self.std=std
        self.gender=gender
        self.parent=parent
        self.address=address
        self.contact=contact
    def details(self):
        print("The name of the student is",self.name)
        print("The grade of the student is",self.std)
        print("The gender of the student is",self.gender)
        print("The parent of the student is",self.parent)
        print("The contact of the employee is",self.contact)
        print("The address of the employee is",self.address)
```

In [83]:
```
s = Student('Polly',9,'Female','Mike & Jessi',9898987676,'Chicago')
s.details()
```

```
The name of the student is Polly
The grade of the student is 9
The gender of the student is Female
The parent of the student is Mike & Jessi
The contact of the employee is Chicago
The address of the employee is 9898987676
```

In [84]:
```
class Sub(Student):
    def __init__ (self,name,std,gender,parent,contact,address,maths,eng
,sci):
        super(). __init__ (name,std,gender,parent,contact,address)
        self.maths = maths
        self.eng = eng
        self.sci = sci
    def sub_details(self):
        print("The obtained mark of",self.name,'in Mathematics is',self
.maths)
        print("The obtained mark of",self.name,'in English is',self.eng
)
        print("The obtained mark of",self.name,'in Science is',self.sci
)
```

In [85]:
```
sub1 = Sub('Harry',8,'Male','Bob & Jenila',9876543423,'Hollywood',98,99
,100)
```

In [86]:
```
sub1.details()
```

```
The name of the student is Harry
The grade of the student is 8
The gender of the student is Male
The parent of the student is Bob & Jenila
The contact of the employee is Hollywood
The address of the employee is 9876543423
```

In [87]:
```
sub1.sub_details()
```

```
The obtained mark of Harry in Mathematics is 98
The obtained mark of Harry in English is 99
The obtained mark of Harry in Science is 100
```

In [88]:
```
#Example-03:(Is also an example of Single Inheritance.)
```

In [89]:
```
class Car:
    def __init__ (self,name,color,cost,milage,hp,model):
        self.name = name
        self.color = color
        self.cost = cost
        self.milage = milage
        self.hp = hp
        self.model = model
    def details(self):
        print("The name of the car is",self.name,'.')
        print("The color of the",self.name,"car is",self.color,'.')
        print("The cost of the",self.name,"car is",self.cost,'.')
        print("The milage of the",self.name,"car is",self.milage,'.')
        print("The horse power of the",self.name,"car is",self.hp,'.')
        print("The model of the",self.name,"car is",self.model,'.')
```

In [90]:
```
car1 = Car('Rolls ROyace','Yellow','Rs.80000000','800 Km/Hr.',2500,'RR9
8A')
```

In [91]:
```
car1.details()
```

```
The name of the car is Rolls ROyace .
The color of the Rolls ROyace car is Yellow .
The cost of the Rolls ROyace car is Rs.80000000 .
The milage of the Rolls ROyace car is 800 Km/Hr. .
The horse power of the Rolls ROyace car is 2500 .
The model of the Rolls ROyace car is RR98A .
```

In [92]:
```
class Rolls(Car):
    def __init__ (self,name,color,cost,milage,hp,model,seat,wheel,brake
):
        super(). __init__ (name,color,cost,milage,hp,model)
        self.seat = seat
        self.wheel = wheel
        self.brake = brake
    def details_Royace(self):
        print("There are",self.seat,'seats in my',self.name,'car.')
        print("There are",self.wheel,'wheels in my',self.name,'car.')
        print("The brake of my",self.name,"car is",self.brake,'.')
```

In [93]:
```
Rolls1 = Rolls("Rolls Royace",'White-Green','Rs.250000000','1600 Km/Hr'
,4800,'AXP998Q',12,8,'Powerbrake')
```

In [94]:
```
Rolls1.details()
```

```
The name of the car is Rolls Royace .
The color of the Rolls Royace car is White-Green .
The cost of the Rolls Royace car is Rs.250000000 .
The milage of the Rolls Royace car is 1600 Km/Hr .
The horse power of the Rolls Royace car is 4800 .
The model of the Rolls Royace car is AXP998Q .
```

In [95]:
```
Rolls1.details_Royace()
```

```
There are 12 seats in my Rolls Royace car.
There are 8 wheels in my Rolls Royace car.
The brake of my Rolls Royace car is Powerbrake .
```

In [96]:
```
#B.Inheritance:-It helps to inherit the properties of one or more class
(s) to another class.
```

In [97]:
```
#Types of inheritance:- i) Single, ii) Multiple, iii) Multi-level and i
v) Hybrid
```

In [98]:
```
#a.Single Inheritance: In this, child class inherits the properties fro
m a single parent class.
```

In [99]:
```
#Example-01:
```

In [100]:
```
class Vehicle:
    def __init__ (self,name,color,cost,seat,bulb):
        self.name = name
        self.color = color
        self.cost = cost
        self.seat = seat
        self.bulb = bulb
    def details(self):
        print("The name of the vehicle is",self.name)
        print('The color of the',self.name,'is',self.color)
        print('The cost of the',self.name,'is',self.cost)
        print('The number of seats in the',self.name,'is',self.seat)
        print('The number of bulbs in the',self.name,'is',self.bulb)
        print("I am a vehicle.")
```

In [101]:
```
v1 = Vehicle('truck','yelloow',5000000,8,25)
v1.details()
```

```
The name of the vehicle is truck
The color of the truck is yelloow
The cost of the truck is 5000000
The number of seats in the truck is 8
The number of bulbs in the truck is 25
I am a vehicle.
```

In [102]:
```
class Bus(Vehicle):
    def set_hp(self,hp):
        self.hp = hp
    def show_hp(self):
        print("The horse power of the", self.name,"is",self.hp)
    def set_wheels(self,wheels):
        self.wheels = wheels
    def show_wheels(self):
        print("The number of the wheels of the",self.name,"is",self.whe
els)
    def drive(self):
        print('Drive me')
```

In [103]:
```
b1 = Bus('bus',"green",6000000,40,80)
b1.set_hp(900)
b1.set_wheels(18)
```

In [104]:
```
b1.details()
```

```
The name of the vehicle is bus
The color of the bus is green
The cost of the bus is 6000000
The number of seats in the bus is 40
The number of bulbs in the bus is 80
I am a vehicle.
```

In [105]:
```
b1.show_hp()
```

```
The horse power of the bus is 900
```

In [106]:
```
b1.show_wheels()
```

```
The number of the wheels of the bus is 18
```

In [107]:
```
b1.drive()
```

```
Drive me
```

In [108]:
```
#Example-02:
```

In [109]:
```
class House:
    def __init__ (self,name,color,cost,room,bulb):
        self.name = name
        self.color = color
        self.cost = cost
        self.room = room
        self.bulb = bulb
    def details(self):
        print("The name of the house is",self.name)
        print('The color of the',self.name,'is',self.color)
        print('The cost of the',self.name,'is',self.cost)
        print('The number of rooms in the',self.name,'is',self.room)
        print('The number of bulbs in the',self.name,'is',self.bulb)
        print("I am a house.")
```

In [110]:
```
h1 = House("'The Everest Palace'",'pink','Rs.25000000',24,120)
```

In [111]:
```
h1.details()
```

```
The name of the house is 'The Everest Palace'
The color of the 'The Everest Palace' is pink
The cost of the 'The Everest Palace' is Rs.25000000
The number of rooms in the 'The Everest Palace' is 24
The number of bulbs in the 'The Everest Palace' is 120
I am a house.
```

In [112]:
```
class Home(House):
    def show(self):
        print("We reside in 'The Everest Palace.")
```

In [113]:
```
h2 = Home(" 'The Everest Palace' ",'green','Rs.30000000',36,240)
```

In [114]:
```
h2.details()
```

```
The name of the house is  'The Everest Palace'
The color of the  'The Everest Palace'  is green
The cost of the  'The Everest Palace'  is Rs.30000000
The number of rooms in the  'The Everest Palace'  is 36
The number of bulbs in the  'The Everest Palace'  is 240
I am a house.
```

In [115]:
```
h2.show()
```

```
We reside in 'The Everest Palace.
```

In [116]:
```
#Example-03:
```

In [117]:
```
class People:
    def __init__ (self,name,age,height,weight,contact):
        self.name = name
        self.age = age
        self.height = height
        self.weight = weight
        self.contact = contact
    def details(self):
        print("The name of the person is",self.name,'.')
        print('The age of',self.name,'is',self.age,'years.')
        print('The height of',self.name,'is',self.height,'ft.')
        print('The weight of',self.name,'is',self.weight,'kg.')
        print('The contact number of',self.name,'is',self.contact)
        print("I am a gentle man.")
```

In [118]:
```
p1 = People("Brad",29,5.6,78,9876543219)
```

In [119]:
```
p1.details()
```

```
The name of the person is Brad .
The age of Brad is 29 years.
The height of Brad is 5.6 ft.
The weight of Brad is 78 kg.
The contact number of Brad is 9876543219
I am a gentle man.
```

In [120]:
```
class Son(People):
    def father(self):
        print(self.name,'is my father.')
```

In [121]:
```
s1 = Son('Jason',48,6.2,82,9876767540)
```

In [122]:
```
s1.details()
```

```
The name of the person is Jason .
The age of Jason is 48 years.
The height of Jason is 6.2 ft.
The weight of Jason is 82 kg.
The contact number of Jason is 9876767540
I am a gentle man.
```

In [123]:
```
s1.father()
```

```
Jason is my father.
```

In [124]:
```
#b. Multiple Inheritance: In this, child class inherits properties from
more than one parent classes.
```

In [125]:
```
#Example-01:
```

In [126]:
```
class Parent1:
    def input_str1(self,str1):
        self.str1 = str1
    def show_str1(self):
        print("The string of first parent class is",self.str1)
class Parent2:
    def input_str2(self,str2):
        self.str2 = str2
    def show_str2(self):
        print("The string of second parent class is",self.str2)
class Derived(Parent1,Parent2):
    def input_str3(self,str3):
        self.str3 = str3
    def show_str3(self):
        print("The string of derived class is",self.str3)
```

In [127]:
```
d1 = Derived()
```

In [128]:
```
d1.input_str1("'Apple'")
d1.input_str2("'Mango'")
d1.input_str3("'I am inheriting the properties of both first and second
parent classes.'")
```

In [129]:
```
d1.show_str1()
```

```
The string of first parent class is 'Apple'
```

In [130]:
```
d1.show_str2()
```

```
The string of second parent class is 'Mango'
```

In [131]:
```
d1.show_str3()
```

```
The string of derived class is 'I am inheriting the properties of both
```

```
first and second parent classes.'
```

In [132]:
```
#Example-02:
```

In [133]:
```
class Parent1:
    def input_str1(self,str1):
        self.str1 = str1
    def show_str1(self):
        print("The string of first parent class is",self.str1)
class Parent2:
    def input_str2(self,str2):
        self.str2 = str2
    def show_str2(self):
        print("The string of second parent class is",self.str2)
class Parent3:
    def input_str3(self,str3):
        self.str3 = str3
    def show_str3(self):
        print("The string of first parent class is",self.str3)

class Derived(Parent1,Parent2,Parent3):
    def input_str4(self,str4):
        self.str4 = str4
    def show_str4(self):
        print("The string of derived class is",self.str4)
```

In [134]:
```
d2 = Derived()
```

In [135]:
```
d2.input_str1("'Los Angeles'")
d2.input_str2("'California'")
d2.input_str3("'Kentucky'")
d2.input_str4("'I am inheriting the properties of all three parent clas
ses.'")
```

In [136]:
```
d2.show_str1()
```

```
The string of first parent class is 'Los Angeles'
```

In [137]:
```
d2.show_str2()
```
```
The string of second parent class is 'California'
```

In [138]:
```
d2.show_str3()
```
```
The string of first parent class is 'Kentucky'
```

In [139]:
```
d2.show_str4()
```
```
The string of derived class is 'I am inheriting the properties of all t
hree parent classes.'
```

In [140]:
```
#Example-03:
```

In [141]:
```
class Parent1:
    def input_str1(self,str1):
        self.str1 = str1
    def show_str1(self):
        print("The string of first parent class is",self.str1)
class Parent2:
    def input_str2(self,str2):
        self.str2 = str2
    def show_str2(self):
        print("The string of second parent class is",self.str2)
class Parent3:
    def input_str3(self,str3):
        self.str3 = str3
    def show_str3(self):
        print("The string of third parent class is",self.str3)
class Parent4:
    def input_str4(self,str4):
        self.str4 = str4
    def show_str4(self):
        print("The string of fourth parent class is",self.str4)
class Derived(Parent1,Parent2,Parent3,Parent4):
    def input_str5(self,str5):
        self.str5 = str5
    def show_str5(self):
        print("The string of derived class is",self.str5)
```

```
In [142]: d3 = Derived()
```

```
In [143]: d3.input_str1("'Hollywood'")
          d3.input_str2("'Chicago'")
          d3.input_str3("'New York'")
          d3.input_str4("'Columbia'")
          d3.input_str5("'I am inheriting the properties of all four parent class
          es.'")
```

```
In [144]: d3.show_str1()

          The string of first parent class is 'Hollywood'
```

```
In [145]: d3.show_str2()

          The string of second parent class is 'Chicago'
```

```
In [146]: d3.show_str3()

          The string of third parent class is 'New York'
```

```
In [147]: d3.show_str4()

          The string of fourth parent class is 'Columbia'
```

```
In [148]: d3.show_str5()

          The string of derived class is 'I am inheriting the properties of all f
          our parent classes.'
```

```
In [149]: #c. Multi-level Inheritance: In this, parent,child & grandchild relatio
          n exists.
```

In [150]:
```
#Example-01:
```

In [151]:
```
class Parent:
    def input_name(self,name):
        self.name = name
    def show_name(self):
        print("The name of grand-child is",self.name,'.')
class Child(Parent):
    def input_age(self,age):
        self.age = age
    def show_age(self):
        print("The age of grand-child named",self.name,"is",self.age,
'.')
class GrandChild(Child):
    def input_gender(self,gender):
        self.gender = gender
    def show_gender(self):
        print("The gender of grand-child named",self.name,"is",self.gen
der,'.')
```

In [152]:
```
gc = GrandChild()
```

In [153]:
```
gc.input_name("Aaron")
gc.input_age(32)
gc.input_gender('Male')
```

In [154]:
```
gc.show_name()
gc.show_age()
gc.show_gender()
```

```
The name of grand-child is Aaron .
The age of grand-child named Aaron is 32 .
The gender of grand-child named Aaron is Male .
```

In [155]:
```
#Example-02:
```

In [156]:
```
class Parent:
    def input_name(self,name,grade):
        self.grade = grade
        self.name = name
    def show_name(self):
        print("The name of grand-child is",self.name,'.')
        print("The grade of grand-child named",self.name,"is",self.grad
e,'.')
class Child(Parent):
    def input_age(self,age,contact,address):
        self.age = age
        self.contact = contact
        self.address = address
    def show_age(self):
        print("The age of grand-child named",self.name,"is",self.age,
'.')
        print("The contact number of grand-child named",self.name,"is",
self.contact,'.')
        print("The address of grand-child named",self.name,"is",self.ad
dress,'.')
class GrandChild(Child):
    def input_gender(self,gender,result):
        self.gender = gender
        self.result = result
    def show_gender(self):
        print("The gender of grand-child named",self.name,"is",self.gen
der,'.')
        print("The result of grand-child named",self.name,"is",self.res
ult,'.')
```

In [157]:
```
gc2 = GrandChild()
```

In [158]:
```
gc2.input_name("Sharon",9)
gc2.input_age(15,9876543421,'California')
gc2.input_gender('Male','Pass')
```

In [159]:
```
gc2.show_name()
gc2.show_age()
```

```
gc2.show_gender()
```

```
The name of grand-child is Sharon .
The grade of grand-child named Sharon is 9 .
The age of grand-child named Sharon is 15 .
The contact number of grand-child named Sharon is 9876543421 .
The address of grand-child named Sharon is California .
The gender of grand-child named Sharon is Male .
The result of grand-child named Sharon is Pass .
```

In [160]:
```
#Example-03:
```

In [161]:
```
class Parent:
    def input_name(self,name,grade,father,mother):
        self.grade = grade
        self.name = name
        self.father = father
        self.mother = mother
    def show_name(self):
        print("The name of grand-child is",self.name,'.')
        print("The grade of grand-child named",self.name,"is",self.grad
e,'.')
        print("The father's name of grand-child named",self.name,"is",s
elf.father,'.')
        print("The mother's name of grand-child named",self.name,"is",s
elf.mother,'.')

class Child(Parent):
    def input_age(self,age,contact,address,friend,height):
        self.age = age
        self.contact = contact
        self.address = address
        self.friend = friend
        self.height = height
    def show_age(self):
        print("The age of grand-child named",self.name,"is",self.age,
'.')
        print("The contact number of grand-child named",self.name,"is",
self.contact,'.')
```

```
        print("The address of grand-child named",self.name,"is",self.ad
dress,'.')
        print("The best friend of grand-child named",self.name,"is",sel
f.friend,'.')
        print("The height of grand-child named",self.name,"is",self.hei
ght,'.')

class GrandChild(Child):
    def input_gender(self,gender,result,performance):
        self.gender = gender
        self.result = result
        self.performance = performance
    def show_gender(self):
        print("The gender of grand-child named",self.name,"is",self.gen
der,'.')
        print("The result of grand-child named",self.name,"is",self.res
ult,'.')
        print("The performance of grand-child named",self.name,"is",sel
f.performance,'.')
```

In [162]:
```
gc3 = GrandChild()
```

In [163]:
```
gc3.input_name("Andron",10,'Philip','Labiya')
gc3.input_age(14,9849391629,'California','Sonya',6.5)
gc3.input_gender('Male','Pass',"'Better than the best'")
```

In [164]:
```
gc3.show_name()
gc3.show_age()
gc3.show_gender()
```

```
The name of grand-child is Andron .
The grade of grand-child named Andron is 10 .
The father's name of grand-child named Andron is Philip .
The mother's name of grand-child named Andron is Labiya .
The age of grand-child named Andron is 14 .
The contact number of grand-child named Andron is 9849391629 .
The address of grand-child named Andron is California .
The best friend of grand-child named Andron is Sonya .
The height of grand-child named Andron is 6.5 .
The gender of grand-child named Andron is Male .
The result of grand-child named Andron is Pass .
The performance of grand-child named Andron is 'Better than the best' .
```