In [5]:
```
## Name: Ananya Agarwal
## Batch: 3CO14
## Roll No.: 102083036
##Submitted To: Dr. Sharad Saxena

##Sol 1:
class Emp:
    EmpId = 0
    EmpName = None
    Points = 0
    Group = None
    Average_Points = 0


    def __init__(self,EmpId = None, EmpName = None):
        self.EmpId = EmpId
        self.EmpName = EmpName

    def addPoints(self,Points):
        self.Points = self.Points + Points

    def removePoints(self,Points):
        self.Points = self.Points - Points
        if self.Points < 0:
            self.Points = 0

    def computeGroup(self):
        if self.Points <= 100:
            self.Group = "Silver"
        elif self.Points > 100 and self.Points <= 500:
            self.Group = "Gold"
        elif self.Points > 500 and self.Points <= 1000:
            self.Group = "Platinum"
        elif self.Points > 1000:
            self.Group = "Diamond"

    def count_Groupwise(self):
        if self.Group == "Silver":
            lSilver.append(self.EmpName)
        elif self.Group == "Gold":
            lGold.append(self.EmpName)
```

```
        elif self.Group == "Platinum":
            lPlatinum.append(self.EmpName)
        elif self.Group == "Diamond":
            lDiamond.append(self.EmpName)

    def display_details(self):
        print("\nEmployee ID of the employee is: ",self.EmpId)
        print("Name of the employee is: ",self.EmpName)
        print("Points of the employee is: ",self.Points)
        print("Group of the employee is: ",self.Group)

lSilver = []
lGold = []
lPlatinum = []
lDiamond = []


e1 = Emp(1,"Ananya")
e2 = Emp(2,"Vasu")
e3 = Emp(3,"Pooja")

e1.addPoints(5000)
e1.removePoints(4999)
e1.computeGroup()

e2.addPoints(1000)
e2.removePoints(20)
e2.computeGroup()

e3.addPoints(700)
e3.removePoints(2)
e3.computeGroup()

e1.count_Groupwise()
e2.count_Groupwise()
e3.count_Groupwise()

print("Total no. of employees are 3!!")
n=3

print("\nThe employees with group Silver are: ",lSilver)
print("The employees with group Gold are: ",lGold)
```

```
print("The employees with group Platinum are: ",lPlatinum)
print("The employees with group Diamond are: ",lDiamond)

e1.display_details()
e2.display_details()
e3.display_details()


Average_Points = (e1.Points+e2.Points+e3.Points)/n
print("\nAverage Points of the 3 employees are: ",Average_Points)
```

```
Total no. of employees are 3!!

The employees with group Silver are:  ['Ananya']
The employees with group Gold are:  []
The employees with group Platinum are:  ['Vasu', 'Pooja']
The employees with group Diamond are:  []

Employee ID of the employee is:  1
Name of the employee is:  Ananya
Points of the employee is:  1
Group of the employee is:  Silver

Employee ID of the employee is:  2
Name of the employee is:  Vasu
Points of the employee is:  980
Group of the employee is:  Platinum

Employee ID of the employee is:  3
Name of the employee is:  Pooja
Points of the employee is:  698
Group of the employee is:  Platinum

Average Points of the 3 employees are:  559.6666666666666
```

In [2]:
```
##Sol 2:
class Property:
    square_footage = 1500
    no_bedrooms = 4
    no_bathrooms = 4

    def __init__(self,square_footage = 1200, no_bedrooms = 3, no_bathrooms = 3):
        self.sqaure_footage = square_footage
        self.no_bedrooms = no_bedrooms
        self.no_bathrooms = no_bathrooms

class House(Property):
    no_stories = 2
    garage = None
    yard_fenced = False

    def __init__(self,no_stories = 3, garage = "Attached", yard_fenced = True):
        self.no_stories = no_stories
        self.garage = garage
        self.yard_fenced = yard_fenced

    def display_House(self):
        print("\nFollowing are the details of the property house: ")
        print("Square footage of the house: ",self.square_footage)
        print("No. of bedroooms in the house: ",self.no_bedrooms)
        print("No. of bathroooms in the house: ",self.no_bathrooms)
        print("No. of stories in the house: ",self.no_stories)
        print("Type of garage in the house: ",self.garage)
        print("Is the yard of the house fenced or not: ",self.yard_fenced)

class Apartment(Property):
    balcony = True
    laundry = "coin"

    def __init__(self,balcony = False, laundry = "en-suite"):
        self.balcony= balcony
        self.laundry = laundry

    def display_Apartment(self):
        print("\nFollowing are the details of the property Apartment: ")
        print("Square footage of the Apartment: ",self.square_footage)
```

```
        print("No. of bedroooms in the Apartment: ",self.no_bedrooms)
        print("No. of bathroooms in the Apartment: ",self.no_bathrooms)
        print("Is there a balcony present in the Apartment: ",self.balcony)
        print("Type of laundry in the Apartment: ",self.laundry)

class Rental:
    rent_cost = 20000
    is_furnished = True
    utility_included = False

    def __init__(self,rent_cost = 15000, is_furnished = False, utility_included = True):
        self.rent_cost = rent_cost
        self.is_furnished = is_furnished
        self.utility_included = utility_included

    def display_rental(self):
        print("\nFollowing are the details of the properties being rented: ")
        print("Rent per month is: ",self.rent_cost)
        print("Is the property furnished or not: ",self.is_furnished)
        print("Are the utilities included or not: ",self.utility_included)

class HouseRental(House,Rental):
    pass

class ApartmentRental(Apartment,Rental):
    pass

class Purchase:
    purchase_price = 1000000
    annual_tax = 20000

    def __init__(self,purchase_price = 10000000, annual_tax = 1000000):
        self.purchase_price = purchase_price
        self.annual_tax = annual_tax

    def display_purchase(self):
        print("\nFollowing are the details of the properties being purchased: ")
        print("Purchase price is: ",self.purchase_price)
        print("estimated annual property taxes are: ",self.annual_tax)

class HousePurchase(House,Purchase):
    pass
```

```
class ApartmentPurchase(Apartment,Purchase):
    pass

def insert():

    while True:
        print("\nMenu to insert more properties in the agent list: ")
        print("Which class object do you want to create: ")

        print("Press 1 to add more properties(Houses) in the agent which are of type rented: ")
        print("Press 2 to add more properties(Houses) in the agent which are of type purchased: ")
        print("Press 3 to add more properties(Apartments) in the agent which are of type rented: ")
        print("Press 4 to add more properties(Apartments) in the agent which are of type purchased: ")
        print("Press 5 to exit: ")

        option = int(input("Enter your choice: "))

        if option == 1:
            o_hr_1 = HouseRental()
            print("New object of HouseRental class created at memory location: \n",o_hr_1)
            agent.append (o_hr_1)
            print("\nAfter appending one more object of HouseRental type, properties the agent now has are: \n", agent)

        elif option == 2:
            o_hp_1 = HousePurchase()
            print("New object of HousePurchase created at memory location: \n",o_hp_1)
            agent.append (o_hp_1)
            print("\nAfter appending one more object of HousePurchase type, Properties the agent now has are: \n", agent

        elif option == 3:
            o_ar_1 = ApartmentRental()
            print("New object of ApartmentRental created at memory location: \n",o_ar_1)
            agent.append (o_ar_1)
            print("\nAfter appending one more object of AppartmentRental type, Properties the agent now has are: \n", ag

        elif option == 4:
            o_ap_1 = ApartmentPurchase()
            print("New object of ApartmentPurchase created at memory location: \n",o_ap_1)
            agent.append (o_ap_1)
            print("\nAfter appending one more object of ApartmentPurchase type, Properties the agent now has are: \n", a
```

```
        elif option == 5:
            print("\nYou have exited from insertion option. Redirecting you to main menu!!")
            break



o_hr = HouseRental(1500,"Attached",True)
o_hp = HousePurchase(1600,"Dettached",False)
o_ar = ApartmentRental(True,"coin")
o_ap = ApartmentPurchase(False,"en-suite")

agent = [o_hr, o_ar, o_hp, o_ap]

while True:
    print("\nMAIN MENU")
    print("Press 1 to add more objects (houses and apartments on rent or purchase) in the agent: ")
    print("Press 2 display all the data related to house which are to be put on rent: ")
    print("Press 3 display all the data related to house which are to be put for purchase: ")
    print("Press 4 display all the data related to apartment which are to be put on rent: ")
    print("Press 5 display all the data related to apartment which are to be put for purchase: ")
    print("Press 6 to exit: ")

    option = int(input("Enter your choice: "))

    if option == 1:
        insert()

    elif option == 2:
        o_hr.display_House()
        o_hr.display_rental()

    elif option == 3:
        o_hp.display_House()
        o_hp.display_purchase()

    elif option == 4:
        o_ar.display_Apartment()
        o_ar.display_rental()

    elif option == 5:
        o_ap.display_Apartment()
```

```
        o_ap.display_purchase()

    elif option == 6:
        print("\nYou have exited from program. Thank you.")
        break
```

```
MAIN MENU
Press 1 to add more objects (houses and apartments on rent or purchase) in the agent:
Press 2 display all the data related to house which are to be put on rent:
Press 3 display all the data related to house which are to be put for purchase:
Press 4 display all the data related to apartment which are to be put on rent:
Press 5 display all the data related to apartment which are to be put for purchase:
Press 6 to exit:
Enter your choice: 1

Menu to insert more properties in the agent list:
Which class object do you want to create:
Press 1 to add more properties(Houses) in the agent which are of type rented:
Press 2 to add more properties(Houses) in the agent which are of type purchased:
Press 3 to add more properties(Apartments) in the agent which are of type rented:
Press 4 to add more properties(Apartments) in the agent which are of type purchased:
Press 5 to exit:
Enter your choice: 4
New object of ApartmentPurchase created at memory location:
```

In [ ]:

In [ ]:

In [ ]: