Task-01

In [2]:

```
class Student:
    def __init__(self, name='Just a student', dept='nothing'):
        self.__name = name
        self.__department = dept
    def set_department(self, dept):
        self.__department = dept
    def get_name(self):
        return self.__name
        self.__name = name
    def __str__(self):
        return 'Name: '+self.__name+' Department: '+self.__department

class BBA_Student(Student):
    def __init__(self, name='default', dept='BBA'):
        super().__init__(name, dept)

print(BBA_Student())
print(BBA_Student('Humpty Dumpty'))
print(BBA_Student('Little Bo Peep'))
```

```
Name: default Department: BBA
Name: Humpty Dumpty Department: BBA
Name: Little Bo Peep Department: BBA
```

Task-02

In [3]:

```
1
2  class Vehicle:
3      def __init__(self):
4          self.x = 0
5          self.y = 0
6      def moveUp(self):
7          self.y+=1
8      def moveDown(self):
9          self.y-=1
10     def moveRight(self):
11         self.x+=1
12     def moveLeft(self):
13         self.x-=1
14     def __str__(self):
15         return '('+str(self.x)+' , '+str(self.y)+')'
16
17 class Vehicle2010(Vehicle):
18     def __init__(self):
19         super().__init__()
20
21     def moveUpperRight(self):
22         super().moveUp()
23         super().moveRight()
24     def moveUpperLeft(self):
25         super().moveUp()
26         super().moveLeft()
27     def moveLowerRight(self):
28         super().moveDown()
29         super().moveRight()
30     def moveLowerLeft(self):
31         super().moveDown()
32         super().moveLeft()
33     def equals(self, new):
34         if self.x == new.x and self.y == new.y:
35             return True
36         else:
37             return False
38
```

```
print('Part 1')
print('------')
car = Vehicle()
print(car)
car.moveUp()
print(car)
car.moveLeft()
print(car)
car.moveDown()
print(car)
car.moveRight()
print(car)
print('------')
print('Part 2')
print('------')
car1 = Vehicle2010()
print(car1)
car1.moveLowerLeft()
print(car1)
car2 = Vehicle2010()
car2.moveLeft()
print(car1.equals(car2))
car2.moveDown()
print(car1.equals(car2))
```

```
Part 1
------
(0 , 0)
(0 , 1)
(-1 , 1)
(-1 , 0)
(0 , 0)
------
Part 2
------
(0 , 0)
(-1 , -1)
False
True
```

Task- 03

In [4]:

```
class Tournament:
    def __init__(self,name='Default'):
        self.__name = name
    def set_name(self,name):
        self.__name = name
    def get_name(self):
        return self.__name

class Cricket_Tournament(Tournament):
    def __init__(self, name = "Default", nt = 0, type = "No type"):
        super().__init__(name)
        self.nt = nt
        self.type = type
    def detail(self):
        return f"Cricket Tournament Name: {self.get_name()} \nNumber of Team

class Tennis_Tournament(Tournament):
    def __init__(self, name, np):
        super().__init__(name)
        self.np = np
    def detail(self):
        return "Tennis Tournament Name: {} \nNumber of Players: {}".format(s

ct1 = Cricket_Tournament()
print(ct1.detail())
print("----------------------")
ct2 = Cricket_Tournament("IPL",10,"t20")
print(ct2.detail())
print("----------------------")
tt = Tennis_Tournament("Roland Garros",128)
print(tt.detail())
```

```
Cricket Tournament Name: Default
Number of Teams: 0
Type: No type
----------------------
Cricket Tournament Name: IPL
Number of Teams: 10
Type: t20
```

```
----------------------
Tennis Tournament Name: Roland Garros
Number of Players: 128
```

Task-04

In [5]:

```
1
2  class Product:
3      def __init__(self,id, title, price):
4          self.__id = id
5          self. __title = title
6          self. __price = price
7      def get_id_title_price(self):
8          return "ID: "+str(self.__id)+" Title: "+self.__title+ " Price: "+str
9
10 class Book(Product):
11     def __init__(self,id, title, price, nmbr, publisher):
12         super().__init__(id, title, price)
13         self.nmbr = nmbr
14         self.publisher = publisher
15     def printDetail(self):
16         return f"{self.get_id_title_price()} \nISBN: {self.nmbr} Publisher:
17
18 class CD(Product):
19     def __init__(self,id, title, price, band, time, genre):
20         super().__init__(id, title, price)
21         self.band = band
22         self.time = time
23         self.genre = genre
24     def printDetail(self):
25         return "{} \nBand: {} Duration: {}minutes \nGenre: {}".format(self.g
26
27 book = Book(1,"The Alchemist",500,"97806","HarperCollins")
28 print(book.printDetail())
29 print("-----------------------")
30 cd = CD(2,"Shotto",300,"Warfaze",50,"Hard Rock")
31 print(cd.printDetail())
```

```
ID: 1 Title: The Alchemist Price: 500
ISBN: 97806 Publisher: HarperCollins
-----------------------
ID: 2 Title: Shotto Price: 300
Band: Warfaze Duration: 50minutes
Genre: Hard Rock
```

Task- 05

In [6]:

```
1
2  class Animal:
3      def __init__(self,sound):
4          self.__sound = sound
5
6      def makeSound(self):
7          return self.__sound
8
9  class Printer:
10     def printSound(self, a):
11         print(a.makeSound())
12
13 class Dog(Animal):
14     pass
15 class Cat(Animal):
16     pass
17
18 d1 = Dog('bark')
19 c1 = Cat('meow')
20 a1 = Animal('Animal does not make sound')
21 pr = Printer()
22 pr.printSound(a1)
23 pr.printSound(c1)
24 pr.printSound(d1)
```

```
Animal does not make sound
meow
bark
```

Task-06

In [7]:

```
class Shape:
    def __init__(self, name='Default', height=0, base=0):
        self.area = 0
        self.name = name
        self.height = height
        self.base = base
    def get_height_base(self):
        return "Height: "+str(self.height)+", Base: "+str(self.base)

class triangle(Shape):
    def __init__(self, name='Default', height=0, base=0):
        super().__init__(name, height, base)
    def calcArea(self):
        self.area = 0.5 * self.base * self.height
    def printDetail(self):
        return "Shape name: {} \n{} \nArea: {}".format(self.name, self.get_h

class trapezoid(Shape):
    def __init__(self, name='Default', height=0, base=0, side = 0):
        super().__init__(name, height, base)
        self.side = side
    def calcArea(self):
        self.area = 0.5 * (self.base+self.side) * self.height
    def printDetail(self):
        return f"Shape name: {self.name} \n{self.get_height_base()}, Side_A:

tri_default = triangle()
tri_default.calcArea()
print(tri_default.printDetail())
print('--------------------------')
tri = triangle('Triangle', 10, 5)
tri.calcArea()
print(tri.printDetail())
print('--------------------------')
trap = trapezoid('Trapezoid', 10, 6, 4)
trap.calcArea()
print(trap.printDetail())
```

```
Shape name: Default
Height: 0, Base: 0
Area: 0.0
-------------------------
Shape name: Triangle
Height: 10, Base: 5
Area: 25.0
--------------------------
Shape name: Trapezoid
Height: 10, Base: 6, Side_A: 4
Area: 50.0
```

Task-07

In [8]:

```
class Football:
    def __init__(self, team_name, name, role):
        self.__team = team_name
        self.__name = name
        self.role = role
        self.earning_per_match = 0
    def get_name_team(self):
        return 'Name: '+self.__name+', Team Name: ' +self.__team

class Player(Football):
    def __init__(self, team_name, name, role, tg, tp):
        super().__init__(team_name, name, role)
        self.tg = tg
        self.tp = tp
        self.gr = 0
    def calculate_ratio(self):
        self.gr = self.tg/self.tp
        self.earning_per_match = (self.tg * 1000) + (self.tp * 10)
    def print_details(self):
        print(f"{self.get_name_team()}"
              f"\nTeam Role: {self.role}"
              f"\nTotal Goal: {self.tg}, Total Played: {self.tp}"
              f"\nGoal Ratio: {self.gr}"
              f"\nMatch Earning: {self.earning_per_match}K")

class Manager(Football):
    def __init__(self, team_name, name, role, win):
        super().__init__(team_name, name, role)
        self.win = win
        self.me = self.win * 1000
    def print_details(self):
        print("{}\nTeam Role: {}\nTotal Win: {}\nMatch Earning: {}K".format(s

player_one = Player('Juventus', 'Ronaldo', 'Striker', 25, 32)
player_one.calculate_ratio()
player_one.print_details()
print('----------------------------------------')
```

```
39 manager_one = Manager('Real Madrid', 'Zidane', 'Manager', 25)
40 manager_one.print_details()
```

```
Name: Ronaldo, Team Name: Juventus
Team Role: Striker
Total Goal: 25, Total Played: 32
Goal Ratio: 0.78125
Match Earning: 25320K
----------------------------------------
Name: Zidane, Team Name: Real Madrid
Team Role: Manager
Total Win: 25
Match Earning: 25000K
```

In [ ]:

```
1
```