(7.4) Aggregation

without Inheritance accessing attributes or properties of another class by storing object of another

Problem 1. Student and Marks

In [1]:
```
class Student:
    def __init__(self, name, age, batch, ph_no):
        self.name = name
        self.age = age
        self.batch = batch
        self.ph_no = ph_no
    def get_details(self):
        print(f"Name: {self.name}")
        print(f"Age: {self.age}")
        print(f"Batch: {self.batch}")
        print(f"Phone Number: {self.ph_no}")
    def __repr__(self):
        return self.name
```

In [2]:
```
s1 = Student('Pankaj', 20, 'CS-B', 8102090648)
print(s1)
```

```
Pankaj
```

In [3]:
```
class Marks:
    __students = []
    def __init__(self, st, maths, chem, phy):
        self.student = st # trapping an object of student class into
        Marks.__students.append(self)
        # object of marks class this relationship is known aggregation
        self.maths = maths
        self.chemistry = chem
        self.physics = phy
    def get_percentage(self):
        per = round((self.maths+self.physics+self.chemistry) / 3 , 2)
        return per
    def get_marks(self):
        print("Maths: ", self.maths)
        print("Chemistry: ", self.chemistry)
        print("Physics: ", self.physics)
    def get_details(self):
        self.student.get_details()
        self.get_marks()
    @classmethod
    def search(cls, name):
        for st in cls.__students:
            if st.student.name.lower().strip() == name.lower().strip():
                st.get_details()
                break
        else:
                print("No such Student Found !!")

    def __str__(self):
        s = f"""
        Name = {self.student.name}
        Percentage = {self.get_percentage()}%
```

```
        """
        return s
    def __del__(self):
        Marks.__students.remove(self)
        del self
```

In [4]:
```
s1 = Marks(Student('Sachin', 24, 'CS-B', 9782131159), 80, 90, 70)
s2 = Marks(Student('Rajat', 34, 'CS-A', 9876543210), 90, 80, 45)
```

In [5]:
```
print(s1)
```

```
        Name = Sachin
        Percentage = 80.0%
```

In [6]:
```
print(s2)
```

```
        Name = Rajat
        Percentage = 71.67%
```

In [7]:
```
s1.get_details()
```

```
Name: Sachin
Age: 24
Batch: CS-B
Phone Number: 9782131159
Maths:  80
Chemistry:  90
Physics:  70
```

In [8]:
```
Marks.search('Rishi')
```

```
No such Student Found !!
```

In [9]:
```
Marks.search("saChin")
```

```
Name: Sachin
Age: 24
Batch: CS-B
Phone Number: 9782131159
Maths:  80
Chemistry:  90
Physics:  70
```

Problem 2. Link List

In [10]:
```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None
```

In [11]:
```
a = Node(20)
b = Node(30)
c = Node(40)
d = Node(50)
```

In [12]:
```
print(a.data)
print(b.data)
print(c.data)
print(d.data)
```

```
20
30
40
50
```

In [13]:
```
a.next = b
b.next = c
c.next = d
```

In [14]:
```
del b
del c
del d
```

In [15]:
```
head = a
```

Traversing of Link List

In [16]:
```
while head.next != None:
    print(head.data)
    head = head.next
else:
    print(head.data)
```

```
20
30
40
50
```

With Aggregation

In [17]:
```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None
```

In [18]:
```
class LinkedList:
    def __init__(self):
        self.head = None
        # self.head --> Node object
    def append(self, node):
        if self.head == None:
            self.head = node
        else:
            temp = self.head
            while temp.next != None:
                temp= temp.next
            else:
                temp.next = node
    def display(self):
        if self.head == None:
            return
        temp = self.head
```

```
        while temp.next != None:
            print(temp.data)
            temp = temp.next
        else:
            print(temp.data)
```

In [19]:
```
a = LinkedList()
a.display()
```

In [20]:
```
a.append(Node(30))
a.append(Node(40))
a.append(Node(50))
a.append(Node(70))

a.display()
```

```
30
40
50
70
```

In [22]:
```
ll = LinkedList()
while input('do you want add record ?') == 'yes':
    data = int(input("Enter Data: "))
    node = Node(data)
    ll.append(node)
```

```
do you want add record ?yes
Enter Data: 10
do you want add record ?yes
Enter Data: 30
do you want add record ?yes
Enter Data: 50
do you want add record ?no
```

In [23]:
```
ll.display()
```

```
10
30
50
```

In [ ]: