7**. Develop a python program that uses class Student which prompts the user to enter marks in three subjects and calculates total marks, percentage and displays the score card details. [Hint: Use list to store the marks in three subjects and total marks. Use \_init\_() method to initialize name, USN and the lists to store marks and total, Use getMarks() method to read marks into the list, and display() method to display the score card details.]**

from functools import reduce

class Student:

def \_\_init\_\_(self, name, USN):

self.name = name

self.USN = USN

self.list = []

def getMarks(self) -> None:

print("Enter the marks of the student: ")

for i in range(0, 3):

self.list.append(int(input()))

def display(self):

max\_marks = int(input("enter max marks"))

TotalMarks = reduce(lambda a, b: a + b, self.list)

percentage = (TotalMarks / max\_marks) \* 100

print("Name:", self.name)

print("USN", self.USN)

print("Marks: ", \*self.list, sep="\n")

print("max marks", max\_marks)

print("obtained marks", TotalMarks)

print("percentage obtained", percentage)

print("Enter the details of student 1")

name = input("Enter the name: ")

USN = input("Enter the usn: ")

s1 = Student(name, USN)

s1.getMarks()

print("\n")

s1.display()

**8. Write a python program for the following:**

**a. Create a class called time. Its three members all type int should be called hours, minutes and seconds. Write a python program that prompts the user to enter a time values separately. The Program should then store the time in the object and finally printout the total no of seconds represented by this value. Use appropriate member functions.**

class Time:

def \_\_init\_\_(self, hours=0, minutes=0, seconds=0):

self.hours = hours

self.minutes = minutes

self.seconds = seconds

def to\_seconds(self):

return self.hours \* 3600 + self.minutes \* 60 + self.seconds

time = Time(hours=int(input("Enter the hours: ")),

minutes=int(input("Enter the minutes: ")),

seconds=int(input("Enter the seconds: ")))

print("Total seconds:", time.to\_seconds())

**b. Write a python program to create a class called Mylist that shadows a python list: it should overload + operator to append the data to the list. Also provide constructor for your class that takes an existing list.**

class Mylist:

def \_\_init\_\_(self, lst=None):

if lst is None:

self.items = []

else:

self.items = lst

def \_\_add\_\_(self, other):

if isinstance(other, list):

return self.items.extend(other)

elif isinstance(other, (int, str)):

return self.items.append(other)

else:

return NotImplemented

def \_\_repr\_\_(self):

return repr(self.items)

my\_list = Mylist([1, 2, 3])

my\_list + [4, 5, 6]

print(my\_list)

my\_list + 7

print(my\_list)

**c. Write a python program to implement the following using Inheritance**

**Employee**

**\_\_\_\_\_|\_\_\_\_\_**

**| |**

**Clerk Software Engineer**

**|**

**Team Leader**

class Employee:

def \_\_init\_\_(self, name, salary):

self.name = name

self.salary = salary

def get\_details(self):

print("from Employee ->")

return "Name: " + self.name + ", Salary: " + str(self.salary)

class Clerk(Employee):

print("from clerk->")

pass

class SoftwareEngineer(Employee):

def \_\_init\_\_(self, name, salary, programming\_languages):

super().\_\_init\_\_(name, salary)

self.programming\_languages = programming\_languages

def get\_details(self):

print("from software engineer->")

return super().get\_details() + ", Programming Languages: " + self.programming\_languages

class TeamLeader(SoftwareEngineer):

def \_\_init\_\_(self, name, salary, programming\_languages, team\_size):

super().\_\_init\_\_(name, salary, programming\_languages)

self.team\_size = team\_size

def get\_details(self):

print("Team leader->")

return super().get\_details() + ", Team Size: " + str(self.team\_size)

e1 = Employee("John", 5000)

print(e1.get\_details())

c1 = Clerk("Jane", 3000)

print(c1.get\_details())

se1 = SoftwareEngineer("Mike", 6000, "Python, Java")

print(se1.get\_details())

tl1 = TeamLeader("Steve", 8000, "Python, Java, C++", 10)

print(tl1.get\_details())