**TASK SEVEN**

**CLASSES AND OBJECTS**

**1.** Write a program that calculates and prints the value according to the given formula:

Q= Square root of [(2\*C\*D)/H]

Following are the fixed values of C and H:

C is 50.

H is 30.

D is a variable whose values should be input to your program in a comma-separated sequence.

import math

import constant

input\_values = input("Enter the values in a comma-separated sequence:")

split\_values = input\_values.split(',')

for i in split\_values:

    D = int(i)

    Q = math.sqrt((2 \* constant.C \* D) / constant.H)

    print("For D Value of {} result is {}".format(D, Q))

OUTPUT:

Enter the values in a comma-separated sequence:4,7,12

For D Value of 4 result is 3.6514837167011076

For D Value of 7 result is 4.83045891539648

For D Value of 12 result is 6.324555320336759

**2.** Define a class named Shape and its subclass Square. The Square class has an init function which takes length as argument. Both classes have an area function which can print the area of the shape where Shape’s area is 0 by default.

class shape:

    def \_\_init\_\_(self):

        pass

    def area(self):

        print(0)

class square(shape):

    def \_\_init\_\_(self, length):

        self.length = length

        super(square, self).\_\_init\_\_()

    def area(self):

        a = (self.length \* self.length)

        print('The area of a square with a side length of %f is %f' % (self.length, a))

s = square(2)

s.area()

OUTPUT:

The area of a square with a side length of 2.000000 is 4.000000

**3.** Create a class to find three elements that sum to zero from a set of n real numbers

**Input array:** [-25,-10,-7,-3,2,4,8,10]

**Expected output:** [[-10,2,8],[-7,-3,10]]

class triple:

    def triple\_num(self):

        input\_array = input("Enter the numbers:")

        input\_split = input\_array.split(',')

        result = []

        for i in range(len(input\_split)):

            input\_split[i] = int(input\_split[i])

        print(input\_split)

        for i in range(len(input\_split)):

            sum = 0

            for j in range(i + 1, len(input\_split)):

                for z in range(j + 1, len(input\_split)):

                    sum = input\_split[i] + input\_split[j] + input\_split[z]

                    if sum == 0:

                        temp = [input\_split[i], input\_split[j], input\_split[z]]

                        result.append(temp)

                        break

        return result

a = triple()

print(a.triple\_num())

OUTPUT:

Enter the numbers:-25,-10,-7,-3,2,4,8,10

[-25, -10, -7, -3, 2, 4, 8, 10]

[[-10, 2, 8], [-7, -3, 10]]

**4.** Create a Time class and initialize it with hours and minutes.

Create a method **addTime** which should take two Time objects and add them.

E.g.- (2 hour and 50 min)+(1 hr and 20 min) is (4 hr and 10 min)

Create another method **displayTime** which should print the time.

Also create a method **displayMinute** which should display the total minutes in the Time.

E.g.- (1 hr 2 min) should display 62 minute.

class MyTime():

    def \_\_init\_\_(self, hours, minutes):

        self.hours = hours

        self.minutes = minutes

    def add\_time(t1, t2):

        hours = int((t1.minutes + t2.minutes) / 60)

        minutes = t1.minutes + t2.minutes

        if (minutes > 60):

            minutes = (t1.minutes + t2.minutes) % 60

        hours += t1.hours + t2.hours

        newtime = MyTime(hours, minutes)

        return newtime

    def \_\_repr\_\_(self):

        return "Test hours:% s minutes:% s" % (self.hours, self.minutes)

    def display\_time(self):

        print("time in hours:", self.hours, 'hours')

    def display\_mins(self):

        print("time in totalmins:", (self.hours \* 60) + self.minutes, 'minutes')

a = MyTime(2, 50)

b = MyTime(1, 40)

c = MyTime.add\_time(a, b)

print([c])

c.display\_time()

c.display\_mins()

OUTPUT:

[Test hours:4 minutes:30]

time in hours: 4 hours

time in totalmins: 270 minutes

**5.** Write a Person class with an instance variable “age” and a constructor that takes an integer as a parameter. The constructor must assign the integer value to the age variable after confirming the argument passed is not negative; if a negative argument is passed then the constructor should set age to 0 and print “Age is not valid, setting age to 0”. In addition, you must write the following instance methods:

**- yearPasses()** should increase age by the integer value that you are passing inside the function.

**- amIOld()** should perform the following conditional actions:I

* f age is between 0 and <13, print “**You are young**”.
* If age is >=13 and <=19 , print “**You are a teenager**”.
* Otherwise, print “**You are old**”.

**Sample Input** for **amIOld():**

-1

4

10

16

18

64

38

**Expected Output for amIOld():**

Age is not valid, setting age to 0.

You are young.

You are young.

You are a teenager.

You are a teenager.

You are old.

You are old.

Consider the age variable to be set to 38 then:

**Sample Input for yearPasses():** 4

**Expected Output for yearPasses():** 42

class Person:

    age = 0

    def \_\_init\_\_(self, integer):

        if integer < 0:

            print("Age is not valid, setting age to 0")

        else:

            self.age = integer

    def yearPasses(self):

        self.age += 1

    def amIOld(self):

        if self.age == 0 and self.age < 13:

            print("You are young")

        elif self.age >= 13 and self.age <= 19:

            print("You are a teenager")

        else:

            print("You are old")

a= Person(30)

a.amIOld()

OUTPUT:

You are old

**END OF TASK SEVEN**