**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.

from math import sqrt  
  
while True:  
 inputsForD = input("Please Enter Numbers for D Separated by Commas: ")  
 inputsForD = inputsForD.split(',')  
  
 C = 50  
 H = 30  
 values = []  
 for D in inputsForD:  
 Q = round(sqrt(2 \*C\* int(D)/H))  
 values.append(Q)  
 print(values)

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(object):  
 def \_\_init\_\_(self,length):  
 self.length = length  
  
 def area(self):  
 return 0  
  
class square(shape):  
 def \_\_init\_\_(self,length):  
 super(). \_\_init\_\_(length)  
  
 def area(self):  
 return self.length\*self.length  
  
square = square(4)  
print(square.area())

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 py\_solution:  
 def threeSum(self, nums):  
 nums, result, i = sorted(nums), [], 0  
 while i < len(nums) - 2:  
 j, k = i + 1, len(nums) - 1  
 while j < k:  
 if nums[i] + nums[j] + nums[k] < 0:  
 j += 1  
 elif nums[i] + nums[j] + nums[k] > 0:  
 k -= 1  
 else:  
 result.append([nums[i], nums[j], nums[k]])  
 j, k = j + 1, k - 1  
 while j < k and nums[j] == nums[j - 1]:  
 j += 1  
 while j < k and nums[k] == nums[k + 1]:  
 k -= 1  
 i += 1  
 while i < len(nums) - 2 and nums[i] == nums[i - 1]:  
 i += 1  
 return result  
  
print(py\_solution().threeSum([-25, -10, -7, -3, 2, 4, 8, 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 minutes.

class Time(object):  
 def \_\_init\_\_(self,hours,minutes):  
 self.hours = hours  
 self.minutes = minutes  
  
 def addTime(time1,time2):  
 hours = time1.hours + time2.hours  
 hours2minutes = hours \* 60  
 totalminutes = hours2minutes + time1.minutes + time2.minutes  
 totalhours = totalminutes // 60  
 leftover\_minutes = totalminutes % 60  
 return Time(totalhours, leftover\_minutes)  
  
 def displayTime(self):  
 return f'{self.hours} hr(s) and {self.minutes} min(s)'  
  
 def displayMinute(self):  
 return self.hours \* 60 + self.minutes  
  
time1 = Time(2,50)  
time2 = Time(1,20)  
time3 = Time.addTime(time1,time2)

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:

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(object):  
 def \_\_init\_\_(self, age):  
 if age < 0:  
 print("Age is not valid, set to 0")  
 age = 0  
 self.age = age  
  
 def yearpasses (self):  
 self.age += 1  
  
 def amiold(self):  
 if 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")  
  
  
person(-1)  
person(4).amiold()  
person(10).amiold()  
person(16).amiold()  
person(18).amiold()  
person(38).amiold()  
person(64).amiold()