MCT-242 : COMPUTER PROGRAMMING-I

using Python 3.9

Prepared By:

Mr. Muhammad Ahsan Naeem

Pre-Lab 7: Practice Session: CLO 3

Tasks

[1] Write a program that will ask user to enter a 4-digit number (We'll assume that user will actually enter a 4-digit number. Later on, we'll see how to apply check and keep asking for valid entry until user enters a valid number). Your program will show the number of zeros in that 4-digit number. (There can be at maximum 3 zeros in 4-digit number!)

Sample output is:

```
Enter a 4-digit number: 5402
No. of zeros in 5402= 1
```

Another Sample output is:

```
Enter a 4-digit number: 2001
No. of zeros in 2001= 2
```

How to do it:

You already did a task of separating digits of a number using division and remainder-division operations. Here you can initialize a variable, say **`count`**, to store the number of zeros. Initialize this variable to **`0`**. Then separate right most digit of the entered number using remainder division and use **`if`** statement to check whether it is zero or not. If it is zero, increment the variable **`count`** as:

```
count+=1;
```

Repeat it three time and the variable **`count`** will have the number of zeros.

[2] Solution of Quadratic Equation

We know the solution of general quadratic equation $\boxed{ax^2 + bx + c = 0}$ is:

$$x_1, x_2 = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Write a program that will ask user to enter the values of **`a`**,**`b`** and **`c`** of the quadratic equation and the program will display the two roots on the screen.

How to do it:

It is simple to find the roots by coding the above equation but keep these points in mind:

- The two roots of the equation can be complex depending upon **discriminant** (square root term). If we attempt to use **`sqrt()`** function of math module with negative input, we will get an error. We solved this issue in one of previous lab by checking if the input is negative then make it positive, use **`sqrt()`** and append a **`j`** with the result. Same can be done here by computing discriminant and applying check on it.

- Appending a `j` with `sqrt()` result is not a very good idea because if the resulting roots are to be used for further calculations, the appended `j` will not be treated as complex part. Therefore, upgrade the code and use complex data type that we learnt in one of previous lab sessions to store the result in complex data type using `complex(a,b)` function.

Sample output is:

```
This Program will solve Quadratic Equation for you

Enter value of a:1
Enter value of b:4
Enter value of c:1

x1=-0.268
x2=-3.732
```

Another Sample output is:

```
This Program will solve Quadratic Equation for you

Enter value of a:1
Enter value of b:2
Enter value of c:3

x1=(-1+1.414j)
x2=(-1-1.414j)
```

Now add two more features in your code:

i. For a valid quadratic equation, the value of `a` cannot be `0`. If user enters `0` as value of `a`, the program will generate a run-time error while dividing by `2a`. Update the code so that an appropriate message is displayed (e.g. `a` cannot be `0`) if user enters `0` as value of `a`.

ii. The two roots of a quadratic equation can be of four types:

- Real and Different
- Complex
- Pure Imaginary
- Real and Same

Update the code so that it also displays the type of the roots.

[3] Type of Triangle:

Write a program that will ask the user to enter three sides of the triangle and will display if the entered triangle is not valid. If valid, it should display the type of the triangle as described here:

- The program should display the type of the triangle on the basis of its three sides as described here:

    - Equilateral triangle → All three sides are equal
    - Isosceles triangle → Only two sides are equal
    - Scalene triangle → All sides are different
- The program then should also display if the entered triangle is a right-angle tringle. If it is not, the program needs not to display that it is not a right-angle tringle.

[4] Write a program that will ask user to enter day and month (e.g. day=7 and month=3). At output the program will show the next day and month.

Sample output is:

```
Enter the day: 7
Enter the month: 3
The next day is: 8
The next month is: 3
```

Another Sample output is:

```
Enter the day: 30
Enter the month: 4
The next day is: 1
The next month is: 5
```

Another Sample output is:

```
Enter the day: 30
Enter the month: 3
The next day is: 31
The next month is: 3
```

[5] Write a program that will ask user to enter day and month (e.g. day=7 and month=3). At output the program will show the next day and month. But this time also include the possibility of wrong input by user. In that case your program must indicate that the entered date is invalid.