Guidelines and instructions:

- This is a graded assignment, you can collaborate to understand the concepts, but plagiarism is not allowed. You are directed to complete all tasks individually. Any case of plagiarism would result in 0 marks.
- Please follow the coding conventions
    - Indent your code properly
    - Use meaningful variable and method names, mind the usage of *PascalCase* and *camelCase* notation
    - Comment your code properly
- You are requested to
    - Create a new branch named **Assignment-04** in the repository you've created in the first assignment
    - If you have not created your repository then create one private repository on GitHub on this pattern: **Rollnumber-EAD** (e.g., **BITF20M001-EAD**)
    - Send GitHub link of your assignment at this email: bsef20m021@pucit.edu.pk
        - Subject line: '<Roll Number>-Assignment-04' (e.g. BCSF20M001-Assignment-04)
- The submission deadline of this assignment is **Tuesday, 17 October, 2023 10:14:59 AM**
- This is a long assignment but it lets you do the problem solving, use different data structures to implement it
- Please note that the assignments submitted after the deadline will not be entertained

Happy coding 😊

Please write the code for the following statements: (35 marks + 35 marks = 70 marks)

1. Equation solver (1 + 1 + 1 + 1 + 4 + 12 + 15 = 35 marks)
   Write a method **string Solve (string equation)** that takes an equation in string format, and returns its solution in string format.

   a. Add logic of addition in this method. (1 mark)

Example 01:	Example 02:
Input: equation = "122 + 323"	**Input:** equation = "1+234"
Output: "445"	**Output:** "235"
Explanation: 122 + 323 = 445	**Explanation:** 1 + 234 = 235

   b. Add logic of subtraction in this method. (1 mark)

Example 01:	Example 02:
Input: equation = "323 - 121"	**Input:** equation = "1-234"
Output: "202"	**Output:** "-233"
Explanation: 122 - 323 = 202	**Explanation:** 1 - 234 = -233

   c. Add logic of multiplication in this method. (1 mark)

Example 01:	Example 02:
Input: equation = "22 x 45"	**Input:** equation = "3x9"
Output: "990"	**Output:** "27"
Explanation: 220 x 45 = 990	**Explanation:** 3 x 9 = 27

   d. Add logic of division in the method. In case of fractional answer, save it to maximum precision. (1 mark)

Example 01:	Example 02:
Input: equation = "22 / 2"	**Input:** equation = "3/9"
Output: "11"	**Output:** "0.333333"
Explanation: 22 / 2 = 11	**Explanation:** 3 / 9 = 0.33

e. Implement DMAS rule, combine above 4 logics to solve little bit more complex equations. (4 marks)

Example 01:
  Input: equation = "22 / 2 x 34 - 4"
  Output: "370"
  Explanation:
    Step1: 22 / 2 = 11
    Step2: 11 x 34 = 374
    Step3: 374 – 4 = 370
	Example 02:
	  Input: equation = "3x4/9+4"
	  Output: "5"
	  Explanation:
	    Step1: 4 / 9 = 0.44445
	    Step2: 3 x 0.44445 = 1.33334
	    Step3: 1.33334 + 4 = 5.33334

Note: DMAS is the order to solve the mathematical equation, the order is to do **D**ivision first then **M**ultiplication then **A**ddition and then **S**ubtraction. Keep this rule in mind to make your life easy.

f. Add logic to solve equations having brackets in them. The best way is to solve the brackets the first, so that the equation is normalized and easy to solve. (12 marks)

```
Example 01:
  Input: equation = "(1 + 1) – 3 x (44 x 5) / 20"
  Output: "-31"
    Step1: (1 + 1) – 3 x (44 x 5) / 20 ➔ 2 – 3 x (44 x 5) / 20
    Step2: 2 – 3 x (44 x 5) / 20 ➔ 2 – 3 x 220 / 20
    Step3: 2 – 3 x 220 / 20 ➔ 2 - 3 x 11
    Step4: 2 – 3 x 11 ➔ 2 – 33
    Step5: 2 – 33 ➔ -31
```

g. Sanity checks: Add checks in the code to verify if the entered equation is correct or not. If the equation is not correct return the message **"Invalid equation".** (15 marks)

```
Example 01:
  Input: equation = "1 + 2 + 4 + 6 + 8"
  Output: "21"
```

```
Example 02:
  Input: equation = "(1 + 1) 3 + 4 x 5"
  Output: "Invalid equation"
  Explanation: There is no operation defined between (1 + 1) and 3
               "(1 + 1) 3 + 4 x 5". The equation should be clear.
```

```
Example 03:
  Input: equation = "(((1 + 1) x 3) + 4 x 5"
  Output: "Invalid equation"
  Explanation: Closing bracket ')' to first bracket is missing
               "(((1 + 1) x 3) + 4 x 5"
```

```
Example 04:
  Input: equation = "1 + 2 + 3 – – 4"
  Output: "Invalid equation"
  Explanation: Operand is missing between operators "1 + 2 + 3 – – 4"
```

```
Example 05:
  Input: equation = "1 + 2 + 3 –"
  Output: "Invalid equation"
  Explanation: No operand found after subtraction operator in given equation.
               There should always be an operand or mini-equation in brackets
               before and after an operation. "1 + 2 + 3 –"
```

___

***Note*:**

I have broken down this assignment into chunks, so that you can develop it in an iterative way. This is a small example of Scrum framework: Requirement ➔ Analysis ➔ Design ➔ Develop ➔ Test ➔ Maintain ➔ Repeat.
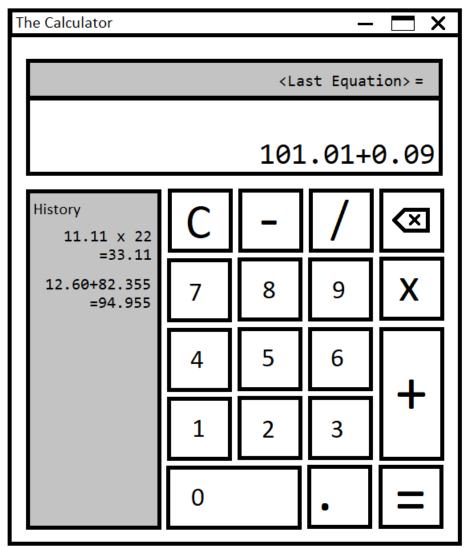
I have given you the Requirements, you are going to do Analysis of it and create a software Design (Flowchart using OOP & DSA concepts) of it, and when the picture is clear you will Develop and Test your work.

So, you can implement all of the logic in a console application, and then continue to the second part of the assignment.

2. The Calculator (35 marks)

Create a Windows Form Application, and create a calculator in it. Your calculator should be able to do the following:

- Write equation in the input **TextBox** using your system keyboard
- Write/update equation in the input **TextBox** using buttons on the calculator pad
- Copy and paste equation in to the input **TextBox**
- Display history of the equation solved in the calculator since the time it started till it is closed
- There should be a clear button to clear the input **TextBox**

The Calculator

‹Last Equation› =

101.01+0.09

History

11.11 x 22
=33.11

12.60+82.355
=94.955

C	-	/	⌫
7	8	9	X
4	5	6	+
1	2	3	+
0	.	=

This is the wireframe of calculator that you are expected to create.

This is a basic design, you can choose design and coloring of the application.

You can see the default calculator in Windows 10, to understand the working of the calculator.

Use the logic of equation solver that you have coded in the first part of the assignment to make the calculator work.

Your calculator should handle the division by zero exception properly.