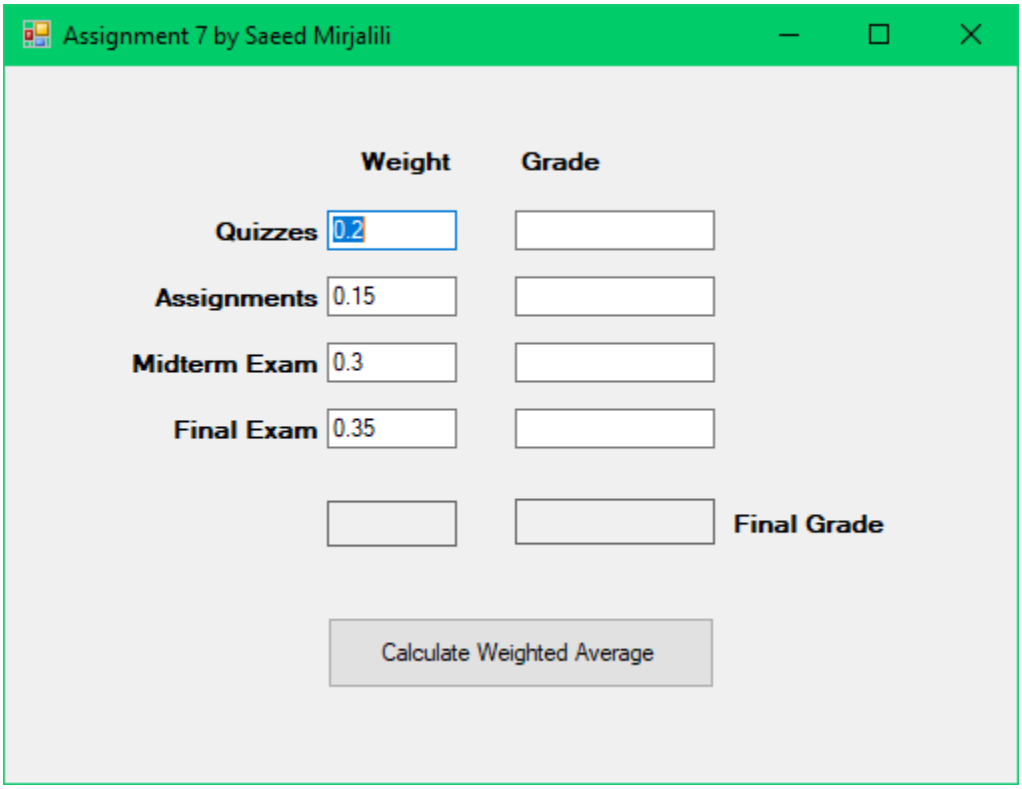
Assignment 7

Step 1 – Create a project of type **Windows Forms App (.Net Framework)**

Step 2 – Design the interface as you can see below



Hint: You only use 9 Labels, 8 Textboxes and 1 Button. The place that your program shows total percentage and Final Grade are labels too.

Step 3 – Give meaningful names to the controls (labels, textboxes and buttons are called a control). Use camelCase style for the name of controls.

Step 4 – Double click on an empty space on the form and a coding window appears for Form1_Load method as below:

```
private void Form1_Load(object sender, EventArgs e)
   {

   }
```

We want to initialize the weights when the form loads. That is why we write our code inside Form1_Load. You must initialize all TextBoxes with a default value for weights. You write code like:

```
quizzesWeight.Text = (0.2).ToString();
```

Note: because `quizzesWeight.Text` is of type string you must convert numbers such as (0.2) to a string before assigning to `quizzesWeight.Text`

Step 5 – Now, we must write our program. When the user clicks on button, it must calculate total percentage (normally this total must be 100) and it must calculate the final grade.

To practice using two dimensional arrays, I ask you to define a two-dimensional array of size 4 x 2 and of type float. We store the values of weights in the first column of the array and the values of the grades in the second column.

Before initializing the array with values, we must convert them from string to float. Use TryParse to convert and if user entered a non-numeric value, your program must show an error message.

Moreover, the values must be between 0 and 100 inclusive.

Additionally, because we do not use Console and we use Graphical User Interface (GUI), we use a MessageBox like below:

```
if (!float.TryParse(quizzesWeight.Text, out studentGrades[0, 0]) // You must complete the code)
     {
       MessageBox.Show("Enter a number between 0 and 1 for Quizzes Weight");
     }

else if (!float.TryParse(assignmentsWeight.Text, // You must complete the code
```

Step 6 – using a for-loop we must calculate the sum of all weights and assign it to the label that we designed for this purpose.

Note: do not forget that you must convert to string before assigning to the label.

Step 7 – using a for-loop, calculate the weighted average of the grades and when the loop finished, assign the result to the label for this purpose.

Marking Scheme

Task	Points	Granted	Comments
Form design is as requested	15		
Meaningful names chosen for controls	8		
Form_load event is working correctly (initialization)	10		
Click_event of the button for calculating the percentage and final grade works correctly	15		
Two dimensional array is used correctly	15		
Converting and validating user inputs to float before storing them in a 2D array	15		
For-loop for calculating the sum	11		
For-loop for calculating the weighted average	11		
Total	**100**	0	