Course: EGDF20
Module: EGE202 Application Programming

Practical 3: Programming WinForm Application using C#.NET

Objectives: At the end of this lab, the student should be familiarized with C# syntax in terms of datatypes & arrays, arithmetic operations, control structures and string operations. Student will also learn about different type of errors like syntax, logic and runtime errors.

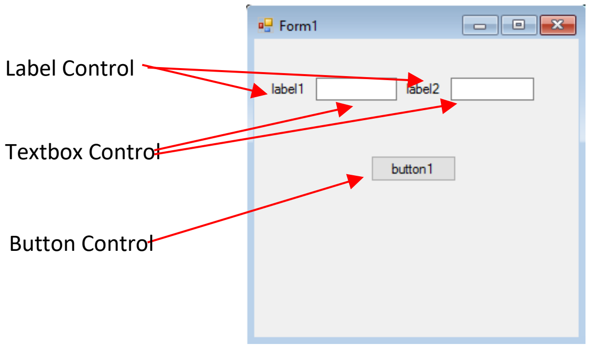
Exercise 1 – Understanding Data Types and Arithmetic Operations

Part 1: Create a WinForm Application to Calculate Square of a number

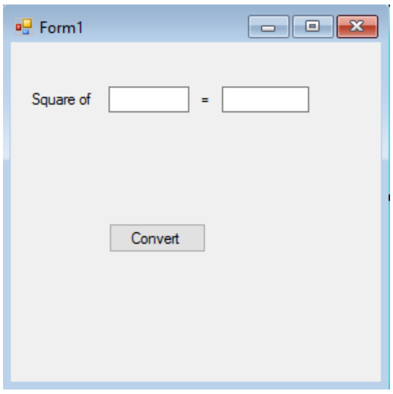
1. Under the ***File*** menu, click ***New Project*** or use the ***New Project*** button to create a new project. Alternatively, use the ***Create New Project*** link in the ***Get Started*** *popup* dialog.
2. From the pop-up dialog, select **"C#"** for the ***Language filter***, **"Windows"** for ***the Platform filter*** and **"Desktop"** for the ***Project type filter***.
3. Then choose ***Windows Forms App (.Net Framework)*** and click the ***Next*** button.
4. Type the name of your new project as ***Converter*** and keep the Solution name the same as Project name.
5. Set the Location to put the project in your own created folder.
6. **Do not** tick on the check-box of [□ **Place solution and project in the same directory**].
7. Click the **Create** button to start your project.

8. In the ***Properties*** window of the ***Form*** control, change the ***TopMost*** property of the ***Form1*** to 'True'.
9. Double click on "Form1.cs" ***Solution Explorer*** window to launch the ***Form Designer*** tab.
10. Change the ***TopMost*** property of the ***Form1*** to 'True'.
11. From the ***Toolbar***, drag in 2 *Textbox*, 2 *Label* and 1 *Button* controls into the *Form1* window.
    - Change the *(Name)* property and *Text* property of these controls based on the table shown below.
    - Arrange these controls as shown in the figure below.
    - Resize ***Form1*** ***(***if necessary)

{Name} From	{Name} To	{Text}
label1	lblSquare	Square of
label2	lblEql	=
textBox1	txtNum	
textBox2	txtResults	
button1	btnConvert	Convert



12. Build and run your application by hitting <F5> or <Ctrl + F5> key. You should see an application with the following UI (user interfacing). Visual Studio through the ***Form Designer*** allows programmer to quickly come up with application user interface while in the subsequent part of the lab, through the ***Code Editor***, programmer can use C# & .NET Framework to develop the logic of the application.

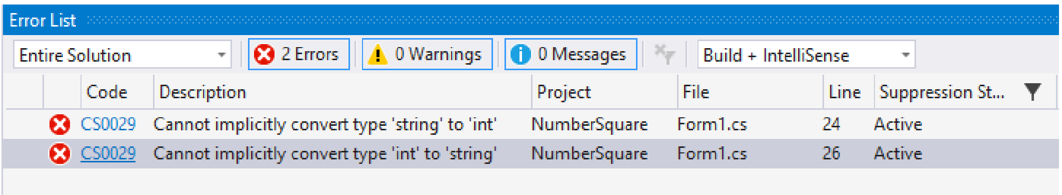


13. Stop the application and proceed to double click on the "Convert" ***Button*** in the ***Form Designer***. That will automatically create *btnConvert_Click (…)* function.

14. Modify *btnConvert_Click (…)* to include the following codes:

```
private void btnConvert_Click(object sender, EventArgs e)
{
    int num, results;

    num = txtNum.Text;
    results = num * num;
    txtResults.Text = results;
}
```

15. Using the ***Build*** menu, select ***Build->Build Solution***. You should expect error to be flagged out at the ***Error List*** window. Double click on the error in the ***Error List*** window and it will bring you to the actual location in the ***Code Editor***.



	Code	Description	Project	File	Line	Suppression St...
⊗	CS0029	Cannot implicitly convert type 'string' to 'int'	NumberSquare	Form1.cs	24	Active
⊗	CS0029	Cannot implicitly convert type 'int' to 'string'	NumberSquare	Form1.cs	26	Active

16. Analyze and explain the codes added in step 14. Further explain and correct the errors.

No	Actions	Explanation
1	`int num, results;`	
	`num = txtNum.Text;`	
	`results = num * num;`	

	txtResults.Text = results;	

2	Explain the nature of the errors found in these codes.
	num = txtNum.Text;
	. . .
	txtResults.Text = results;	

3	**Coding Task:**
	Modify the codes which gave errors. Explain the use of *int.Parse (…) and int.ToString(…)*
	num = int.Parse(txtNum.Text);
	. . .
	txtResults.Text = results.ToString();	

4	Run the application program and try to enter 1.5 as the number to be squared. Is the results expected? Why?	
5	**Coding Task:** Modify the codes to provide a fix to the observation on item 4 above.	

17. Navigate to the ***Form Designer*** tab. Move the mouse over the *Form1* title bar and double click. That will automatically open the “Form1.cs” file. The event handling function *Form1_Load (…)* is automatically appended to the file. The default naming convention for the function is *{control name}_{event type} (…)*
18. Modify *Form1_Load (…)* to include the following codes:

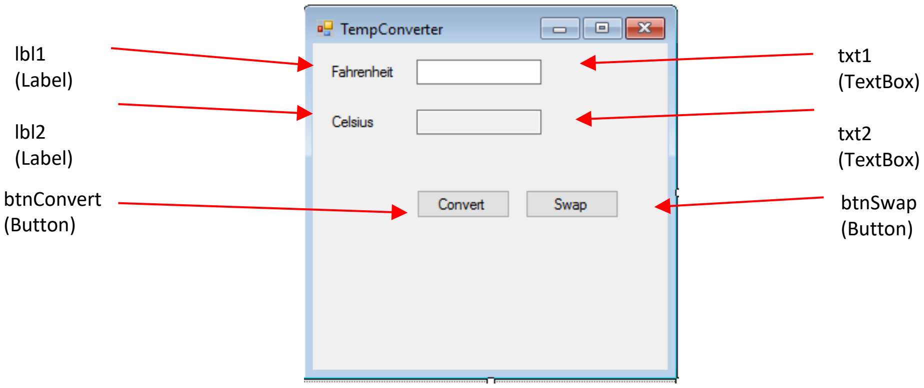
```
private void Form1_Load(object sender, EventArgs e)
{
    txtNum.Text = "0";
    txtResults.Text = "0";
}
```

19. Build and run your application. Observe that now the Textboxes are preloaded with value 0.
20. Under ***File*** menu, click ***Close Solution*** to close your project.

Exercise 2 – Understanding If-Else Control Structure and Complex Arithmetic Operation

Part 1: Create a WinForm Application to Perform Temperature Conversion

1. Download the pre-created **Visual Studio** solution **TempConverter(Student).zip** and unzipped it to your own folder.
2. Under the ***File*** menu, click ***Open -> Project/Solution*** to open the project.
3. Your Form design should look like the figure below.
   Note: The ***ReadOnly*** property for the Celsius *Textbox* is set to *false*. This will prevent user from entering values into that *Textbox* control.



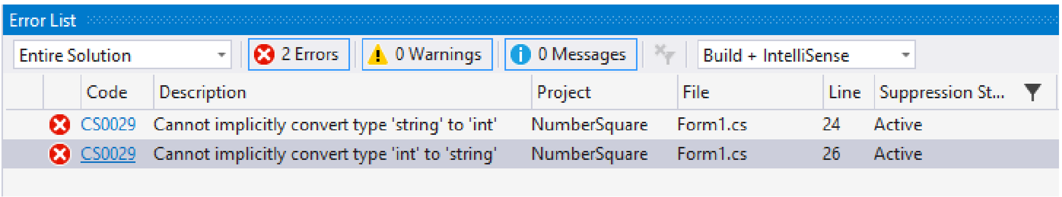
4. Implement the following codes for the button click event for Convert and Swap button.
   - Convert Button: Conversion from Fahrenheit to Celsius or Fahrenheit to Celsius
     Note: You will observe syntax error. Solve it at step 7
   - Swap Button: Switch between Fahrenheit to Celsius or Fahrenheit to Celsius conversion

```
private void btnConvert_Click(object sender, EventArgs e)
{
    double fahr, cels;

    fahr = txt1.Text;
    cels = (fahr - 32) * (5 / 9);
    txt2.Text = cels;
}

private void btnSwap_Click(object sender, EventArgs e)
{
    if (flagF2C == true)
    {
        lbl1.Text = "Celsius";
        lbl2.Text = "Fahrenheit";
        flagF2C = false;
    }
    else
    {
        lbl1.Text = "Fahrenheit";
        lbl2.Text = "Celsius";
        flagF2C = true;
    }
}
```

5. Add the following **bool** type declaration at the beginning of the ***Form1*** class.

```
public partial class Form1 : Form
{
    bool flagF2C = true;

    public Form1()
    {
```

6. Using the ***Build*** menu, select ***Build->Build Solution***. You should expect error to be flagged out at the ***Error List*** window. Double click on the error in the ***Error List*** window and it will bring you to the actual location in the ***Code Editor***.



	Code	Description	Project	File	Line	Suppression St...
☒	CS0029	Cannot implicitly convert type 'string' to 'int'	NumberSquare	Form1.cs	24	Active
☒	CS0029	Cannot implicitly convert type 'int' to 'string'	NumberSquare	Form1.cs	26	Active

7. Analyze and explain the errors observed in step 6. Further explain and correct the errors.

No	Actions	Explanation / Action

1	<u>**Syntax Error**</u>
	Explain the nature of the errors found in these codes and fixed it!
	fahr = txt1.Text;
	. . .
	txt2.Text = cels;	

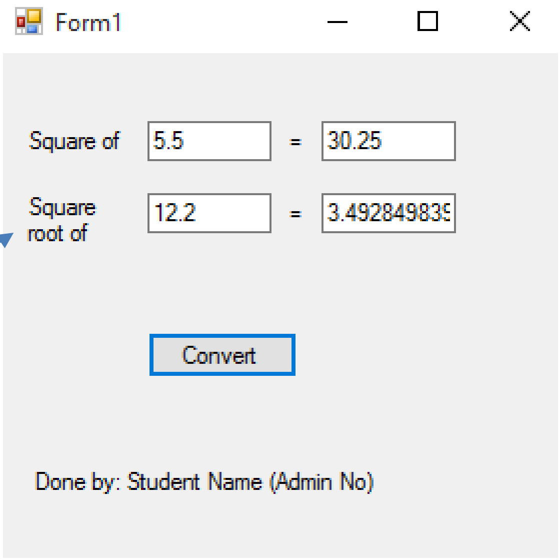
2	<u>**Logic Error**</u> Run the application program and try to enter 100 Fahrenheit. Is the results expected? Why? Fix the error by modifying the codes.	
3	<u>**Runtime Error**</u> Run the application program and click convert straight away without entering any value for Fahrenheit. Is the results expected? Why? Fix the error by adding codes.	

4	Explain what happens when ***Swap*** button is clicked. Hint: Use breakpoint to step through the codes	
5	**Coding Task:** Modify *btnConvert_Click(…)* to include calculation of Celsius to Fahrenheit	

(Exercise) Create a WinForm Application to Calculate Square and Square Root of a number

1. Based on the previous **Converter** solution, modify the application to include square root calculation. The UI design should follow the figure below.
2. Replace the student name and admin no with your own name and admin no.

Hint: Primitive arithmetic operators like ‘+’, ‘-‘, ‘*’, ‘/’, ‘%’ are all supported in .NET. On top of that .NET has a class library called ***Math*** for complex mathematical operations



Square of 5.5 = 30.25

Square root of 12.2 = 3.49284983€

Done by: Student Name (Admin No)

Hint: Set AutoSize to false