Lab Instructions and Solution for Module- 1: Visual Studio Installation, First Project Creation and Details Use of Toolbox

Objectives

After completing this module you will be able to:

* Learn to use Visual Studio IDE 2010 and Installation Process
* How to execute a program in the Visual Studio .Net environment using visual C#
* Basic C# program with "Hello World"
* Use of Toolbox of Visual Studio

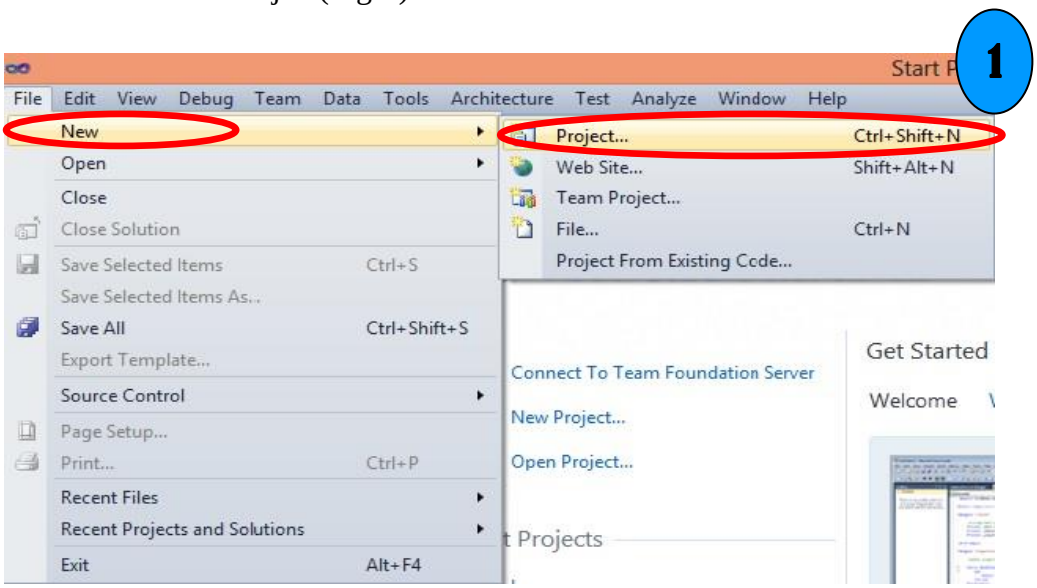
Task 1: Installation process of Visual Studio

Task 2: Basic C# programming with 'Hello World'

At first we will create a very simple first desktop application. In this application, you will click on a button 'Show' and a messagebox (a small popup box) will appear displaying 'Hello World'.

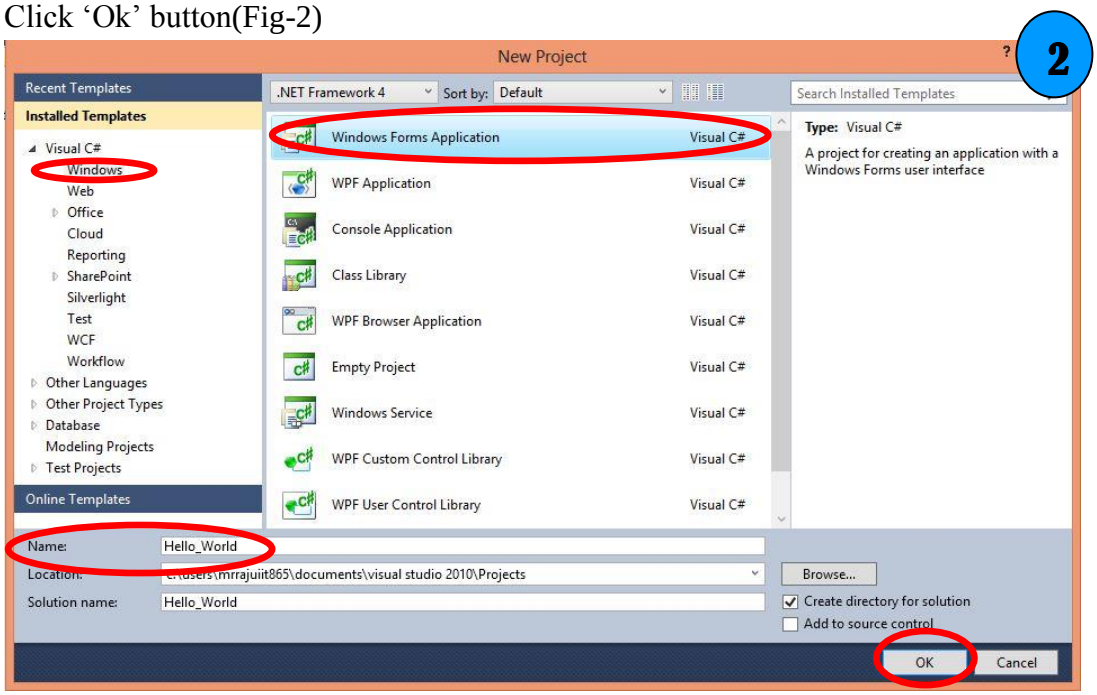
Steps:

1. Start Microsoft Visual Studio 2010
2. Click File>New>Project(Fig-1)

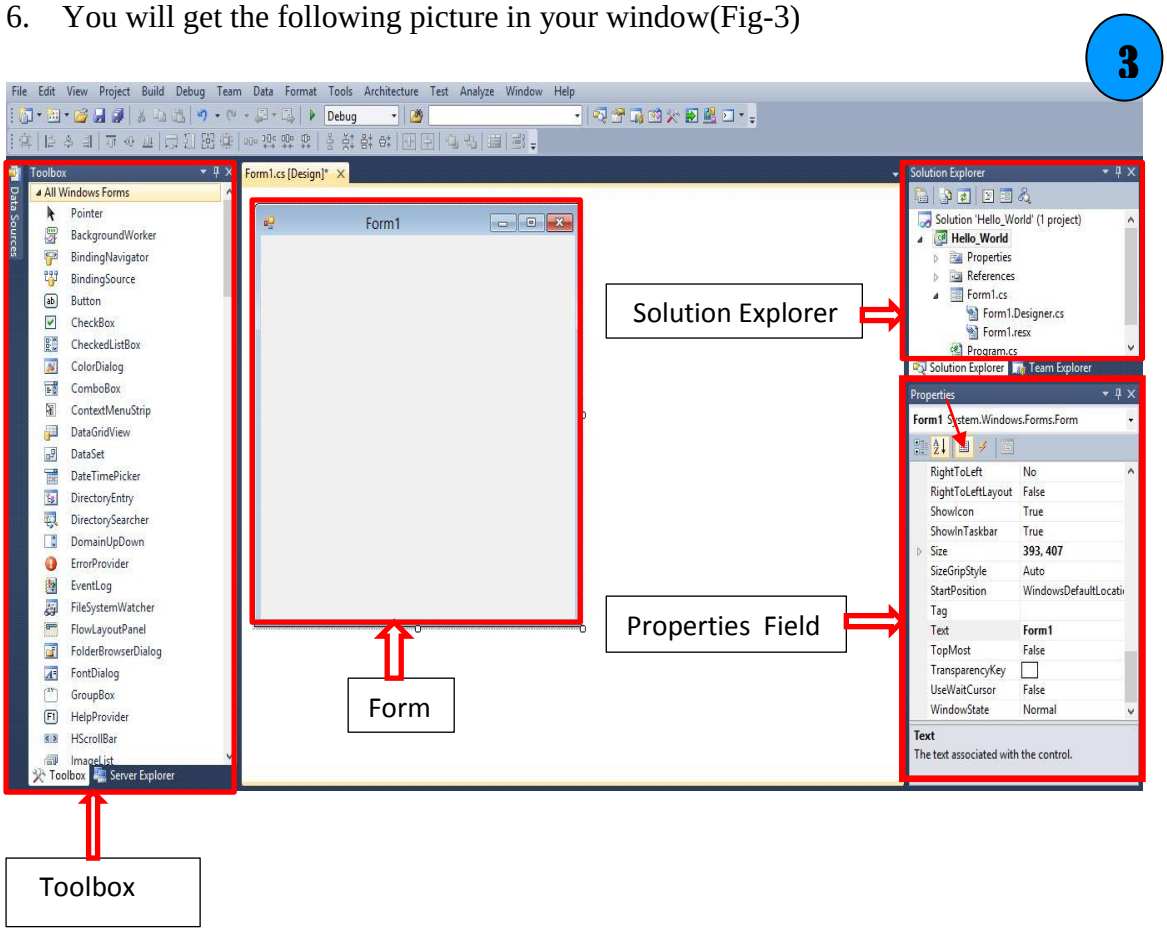


Prepared by:

3. Select Windows>Windows Forms Application(Fig-2)
4. Give an appropriate Name.Here we have given Hello_World(Fig-2).
5. Click 'Ok' button(Fig-2)

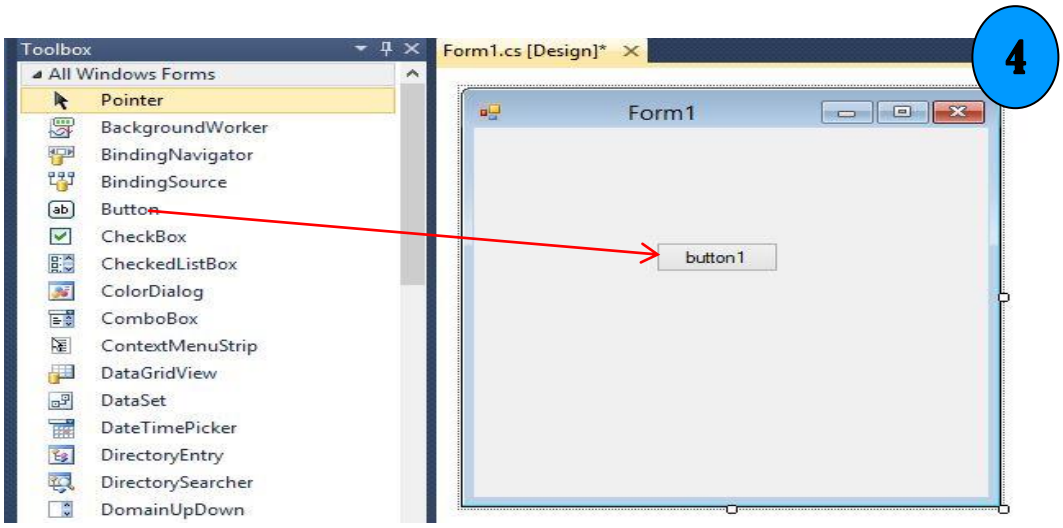


6. You will get the following picture in your window(Fig-3)

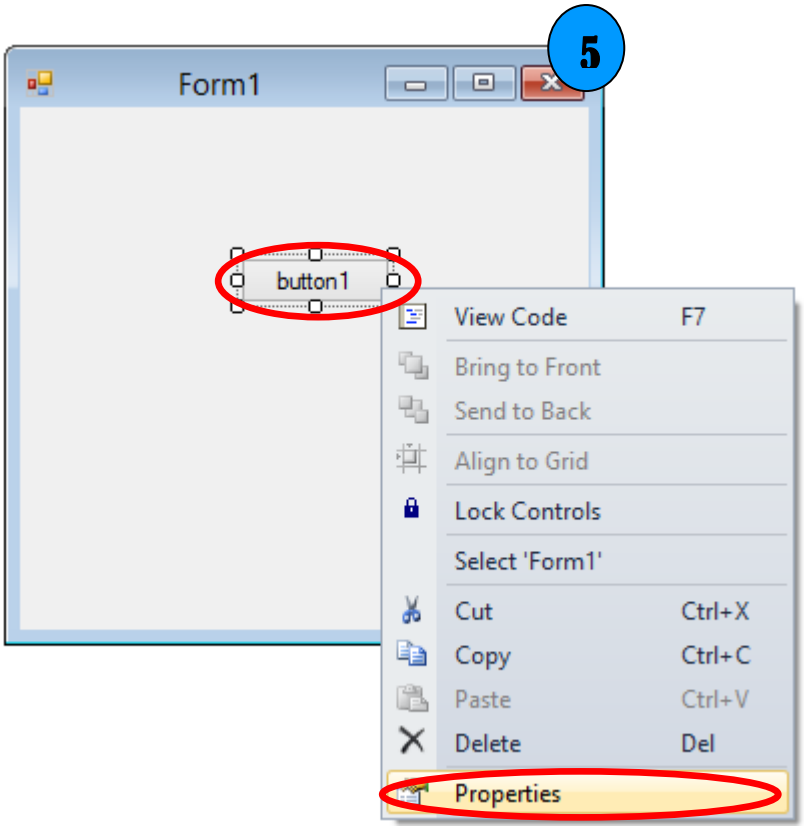


Prepared by:

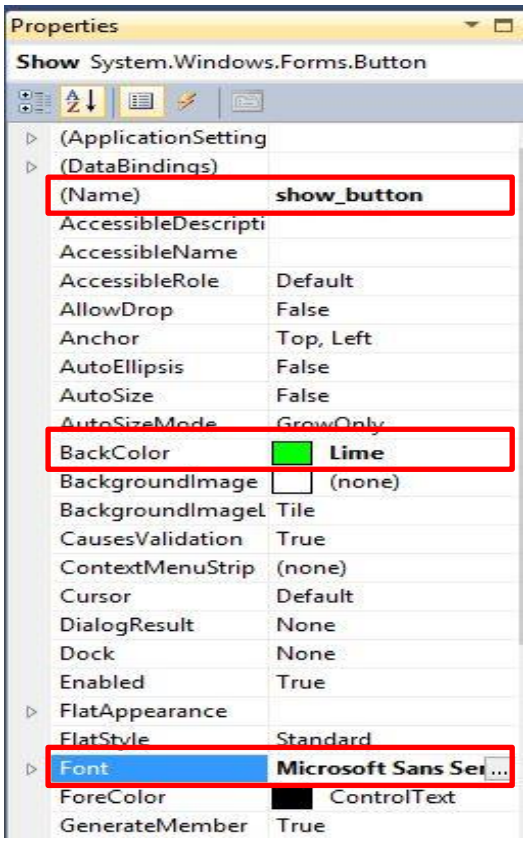
7. Now drag and drop a Button from Toolbox situated at the left side of your development window and you will get the following picture(Fig-4)

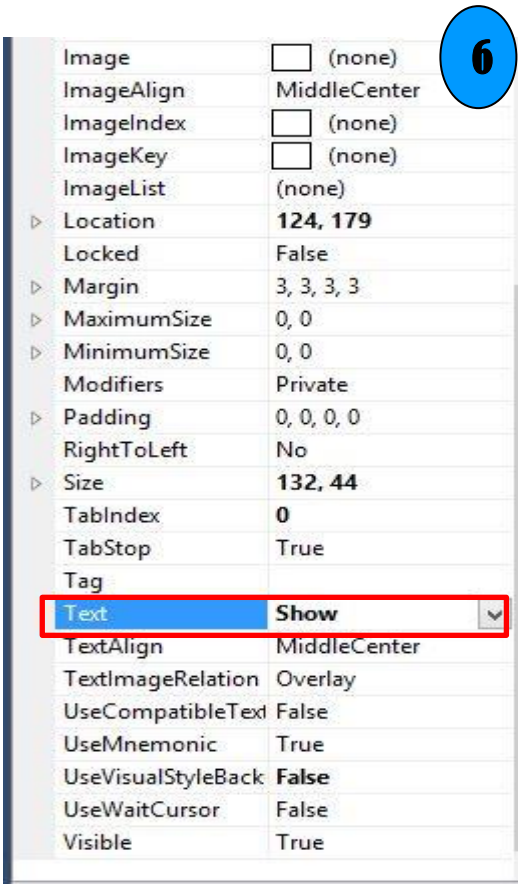


8. Select the button1 and rightclick on the button1 and click the Properties(Fig-5)



9. Change the Name Property,Backcolor,Front and Text Property of the button1 into 'show_button','Lime','Microsoft Sans Serif, 15.75pt, style=Bold',and 'Show'. (According to your choice) and Every time press Enter from your keyboard(Fig-6).



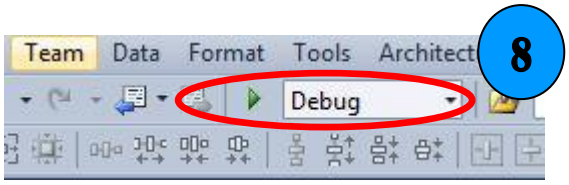


10. After change the properties ,you will get the following picture(Fig-7).

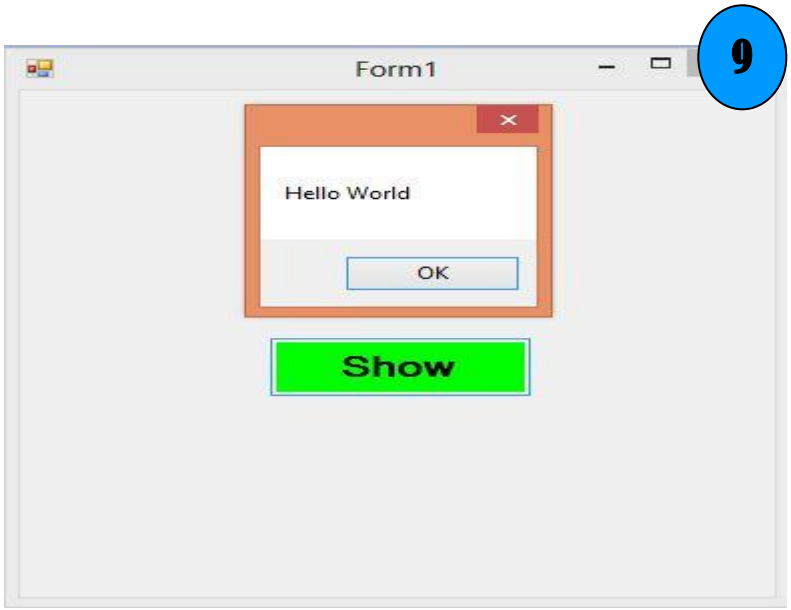
11. Double click on the 'Show' Button of the form and write the following code snippet

```
Private void show_button_Click(object sender, EventArgs e)
        {
MessageBox.Show("Hello World");
        }
```

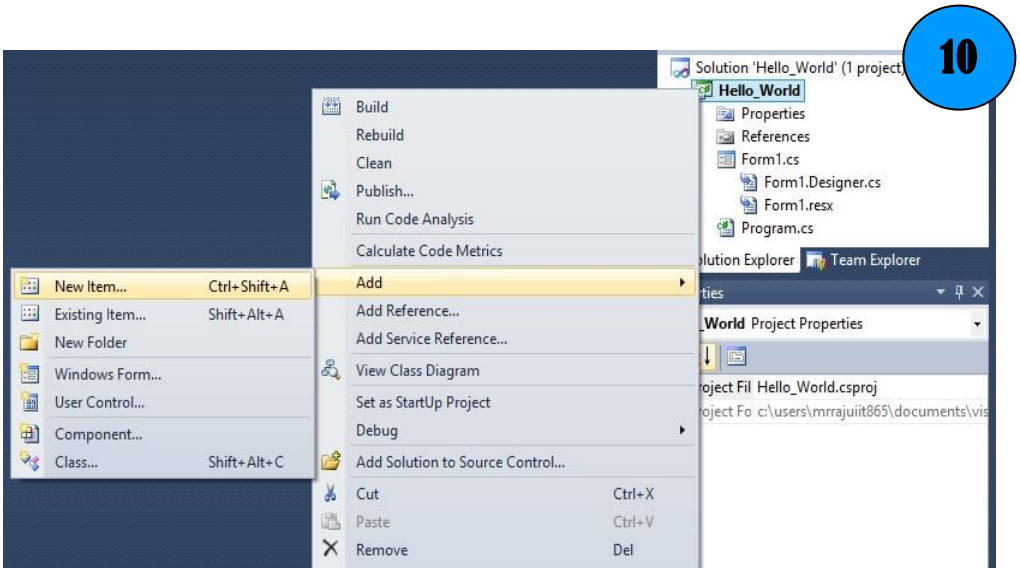
12. Now run the application by clicking on the debug button(Fig-8) or pressing F5



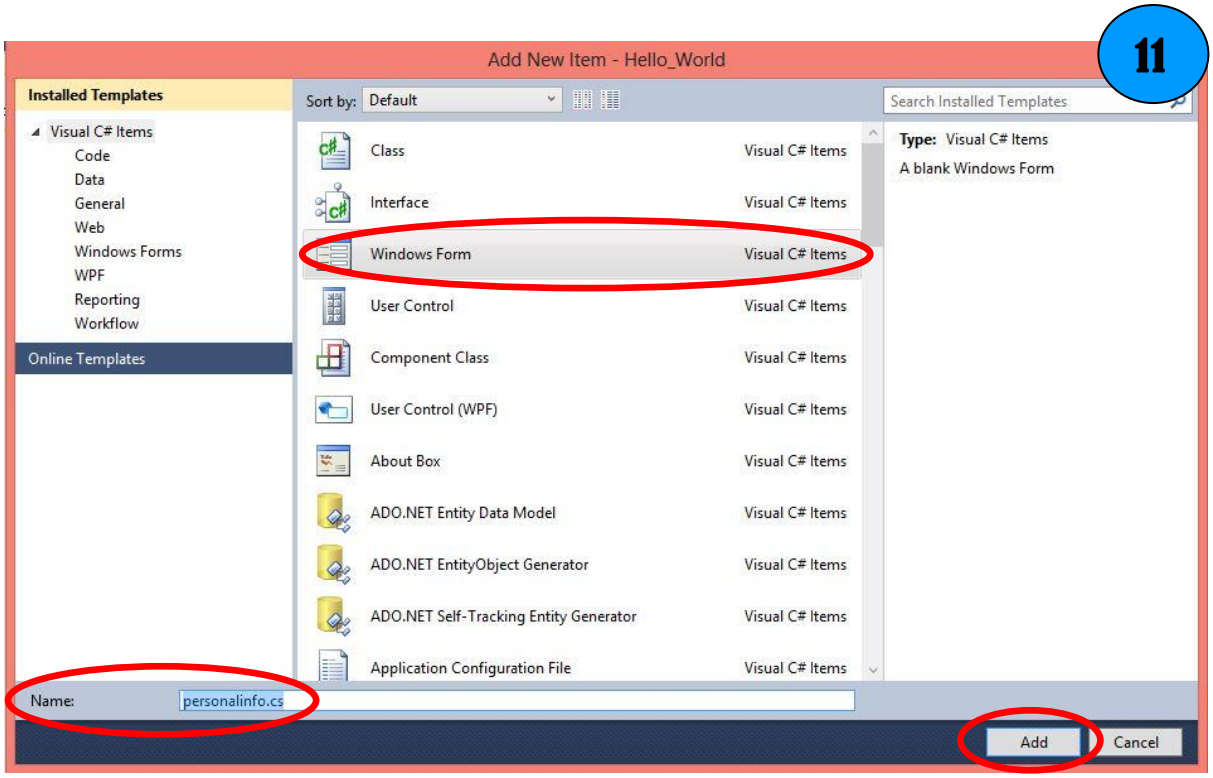
13. Click on the 'Show' Button of the running application(Fig-9)
14. A messagebox will appear. You will get the following picture(Fig-9)



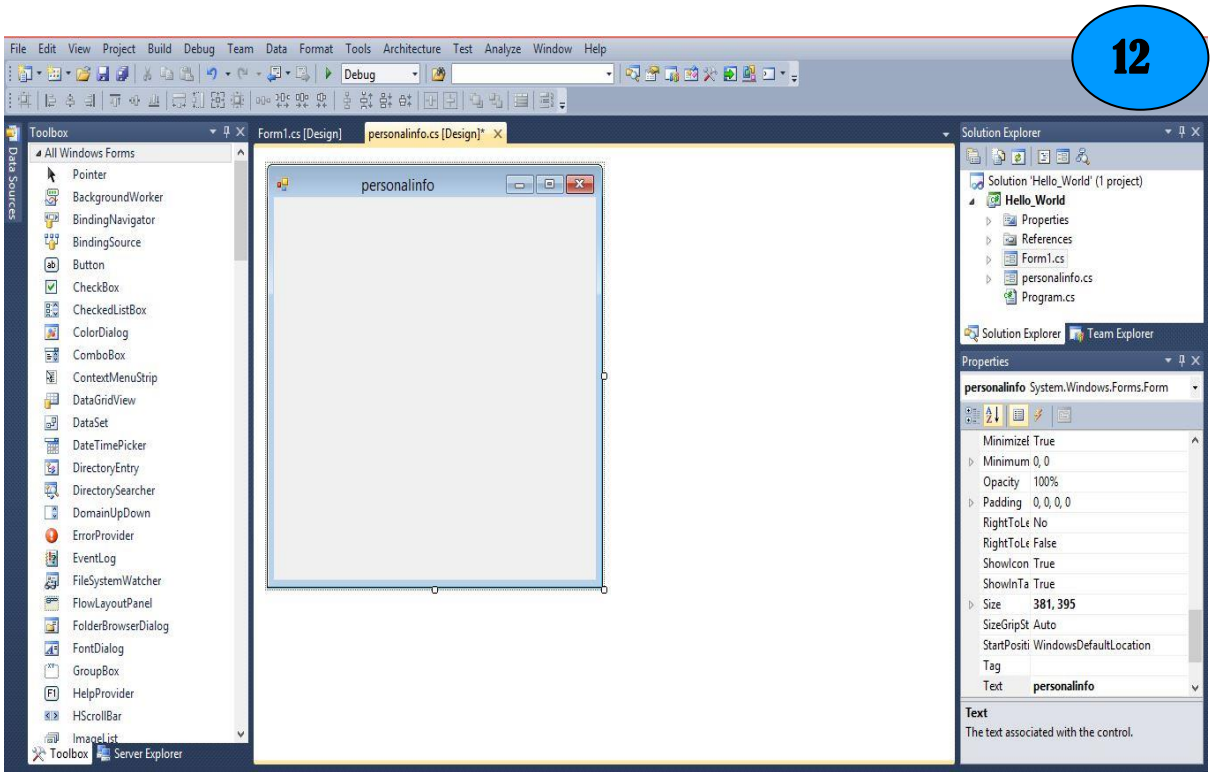
After complete this project if You want to add new item(Class,Windows Form etc.) in this project then You can right click on the project and will get the following picture(Fig-10):



After click the New Item You will get the following picture(Fig-11). Here You select Windows Form and change the Name into personalinfo.cs(Fig-11).



After click the Add button you will get the new windows form in the same project and done another work in this form(Fig-12).



Task 3: C# Program Structure in Desktop Application

Class library:

Like all other programming language, C# also follows some rules. At first, a sample program needs some system library classes.

For example:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
```

-Using System is a statement to indicate that you are using a namespace called System. Basically this namespace contains a class (Console) with methods needed to print the values on the screen and read the values from the keyboard.

-Using System.Collections.Generic is a statement to use Generic namespace that contains interfaces and classes to allow the user to create strong types collections.

-Using System.Data namespace provides access to classes that represent the ADO.NET architecture. ADO.NET lets you build components that efficiently manage data from multiple data sources.

-Using System.Drawing namespace provides access to GDI+ basic graphics functionality. More advanced functionality is provided in the System.Drawing.Drawing2D, System.Drawing.Imaging, and System.Drawing.Text namespaces.

-Using System.Linq is a statement to use Linq namespace that contains interfaces and classes that support query.

-Using System.Text is a statement to use Text namespace that contains classes that represent ASCII, Unicode, UTF-7, and UTF-8 character encodings, convert bytes to characters and vice versa, and manipulate and format string objects.

-Using System.Windows.Forms namespace contains classes for creating Windows-based applications that take full advantage of the rich user interface features available in the Microsoft Windows operating system.

Namespace:

A project can have multiple classes. To organize these classes, multiple folders can be created. The conceptual hierarchy is called the namespace.

For example:

```
namespace Hello_World
```

This is a sample namespace of a project.

Main Method:

Main() method is the entry point of any kind of high level language. C# is not indifferent from that. A C# project must have a Main() method. The code written for a main method is:

```
static void Main(string[] args)
```

When we run an application it starts from Main() method.

Task 4: Use of Toolbox and User Interface Design

Exercise-1: Use of Labels,TextBoxes and Buttons

Labels are one of the most frequently used C# control. We can use the Label control to display text in a set location on the page. Label controls can also be used to add descriptive

text to a Form to provide the user with helpful information.The Label class is defined in the System.Windows.Forms namespace.

A **TextBox** control is used to display, or accept as input, a single line of text.

A **Button** is a control, which is an interactive component that enables users to communicate with an application. A Button can be clicked by using the mouse, ENTER key, or SPACEBAR if the Button has focus.

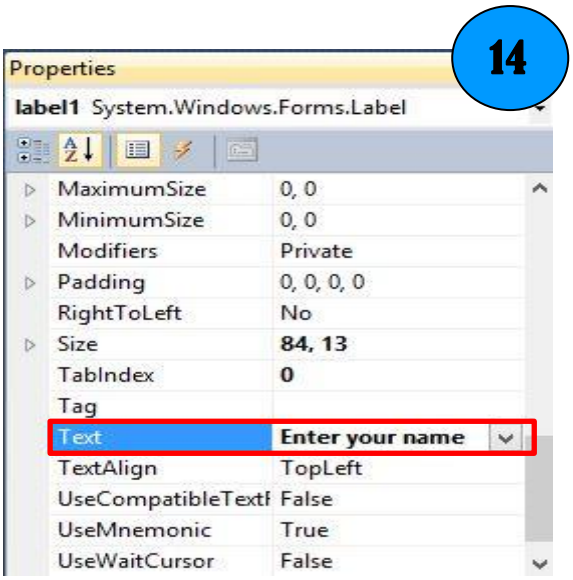
Steps to create a user interface with Label , Textbox and Button controls :

1. Start Microsoft Visual Studio 2010
2. Click File>New>Project
3. Select Windows>Windows Forms Application
4. In the Name box, type TextBoxExample, and then click OK.

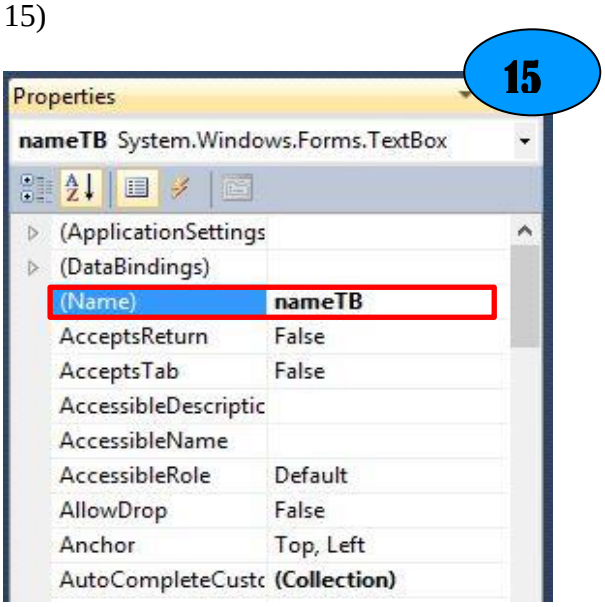
   A new Windows Forms project opens.

5. From the Toolbox, drag a TextBox, Label, and Button control onto the form(Fig-13).

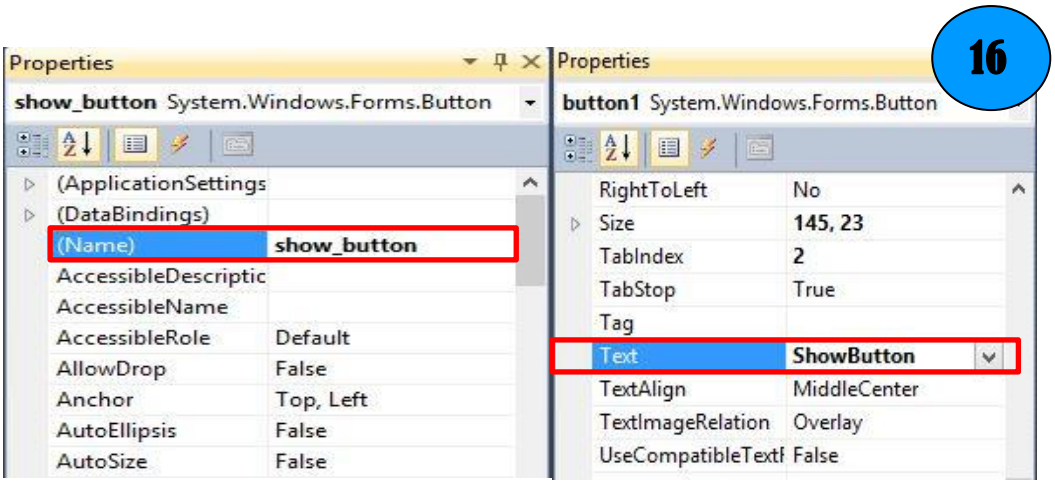
6. Select Label Toolbox and rightclick on the label1 and go to the Properties and change the Text property of the Label control to the following text: Enter your name(Fig-14)



Select TextBox Toolbox and rightclick on the TextBox and go to the Properties and change the Name property of the TextBox control to the following text: nameTB(Fig-15)

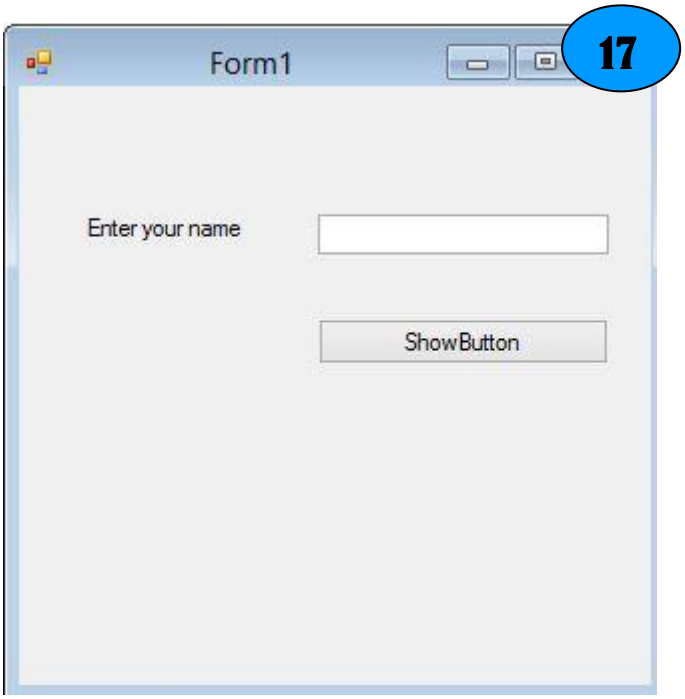


Select Button Toolbox and rightclick on the button1 and go to the Properties window, change the Name property of the Button control to the following text: show_button and change the Text property of the Button control to the following text:ShowButton (Fig-16)



After change property ,every time you must press Enter from your keyboard.

7. After doing this the windows form look like as(Fig-17):



Now that you have created a basic user interface, you will have to add a bit of code to your program, and then it will be ready to test.

To add code and test your program

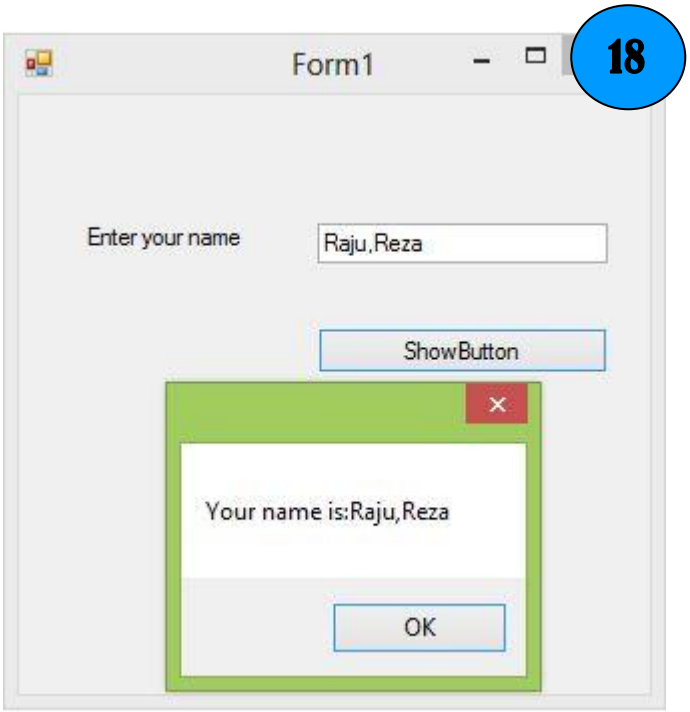
8. Double-click the ShowButton to open the Code Editor.

   The Code Editor opens the `show_button_Click` event handler.

9. Add the following line of code to the `show_button_Click` event handler.

```
private void show_button_Click(object sender, EventArgs e)
        {
            String name = nameTB.Text;
            MessageBox.Show("Your name is:" + name);
        }
```

10. Press F5 to run your program.
11. When the form appears, type your name in the TextBox control and click the ShowButton. A message box appears that displays the text in the TextBox control. Change the text and click the button again. Each time that you click the button, the updated text is displayed.You will get the following picture(Fig-18):

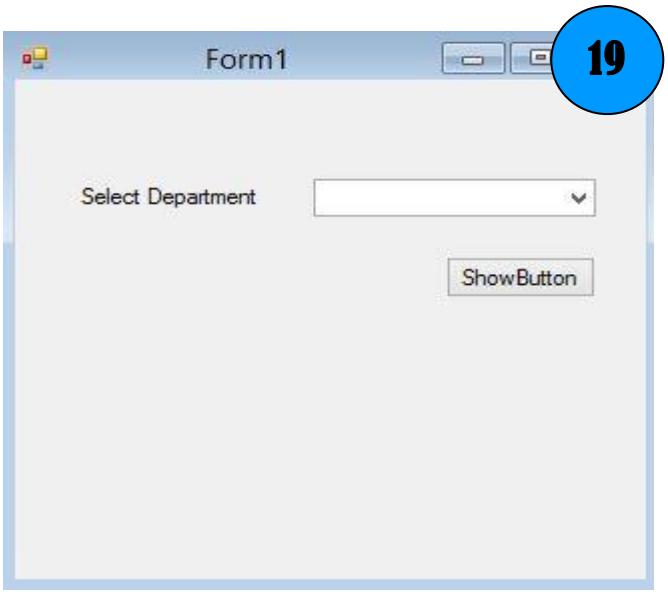


Exercise-2: Use of ComboBox control

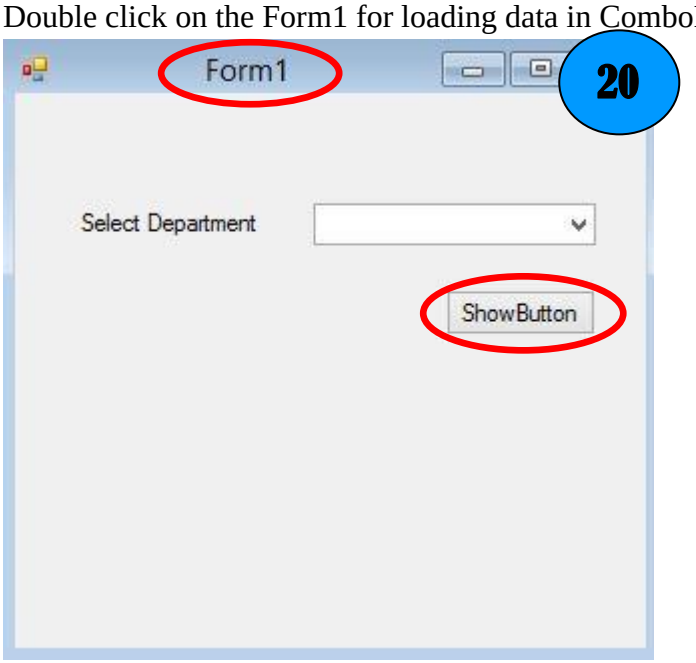
A **ComboBox** displays a text box combined with a ListBox, which enables the user to select items from the list or enter a new value.

Steps:

1. Create a desktop application and change it's properties(As like as Exercise-1)
2. Design the user interface like the following figure(Fig-19)



3. Double click on the Form1 for loading data in ComboBox(Fig-20).



4. Write down the code snippet in the block

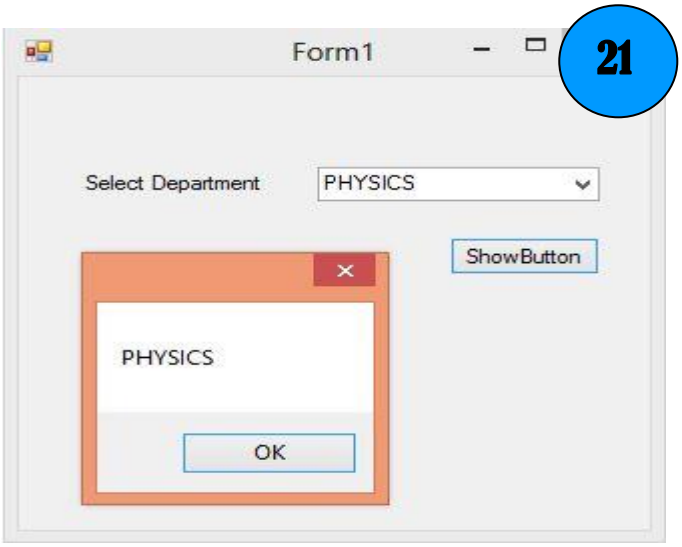
```
private void Form1_Load(object sender, EventArgs e)
{
    dnameCB.Items.Add("IIT");
    dnameCB.Items.Add("CSE");
    dnameCB.Items.Add("URP");
    dnameCB.Items.Add("MATH");
    dnameCB.Items.Add("PHARMACY");
    dnameCB.Items.Add("PHYSICS");
    dnameCB.Items.Add("STATISTICS");
    dnameCB.SelectedIndex = dnameCB.FindStringExact("IIT");

}
```

5. Double click on the ShowButton(Fig-20)

6. Write down the code snippet in the block

```
private void show_button_Click(object sender, EventArgs e)
{
    string dname = dnameCB.Text;
    MessageBox.Show(dname);
}
```

7. Build the solution and run it and You will get the following picture(Fig-21)

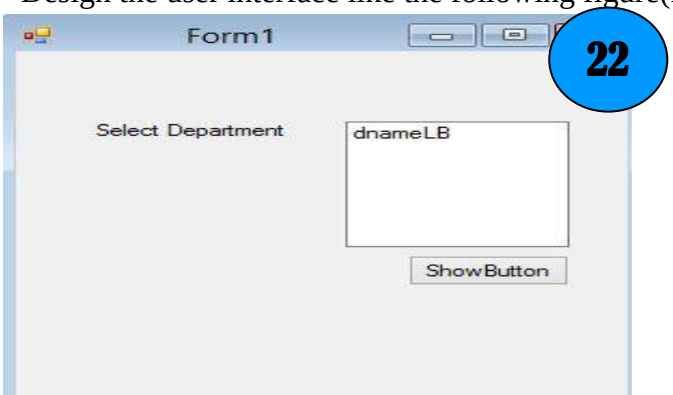


Exercise-3: Use of ListBox control

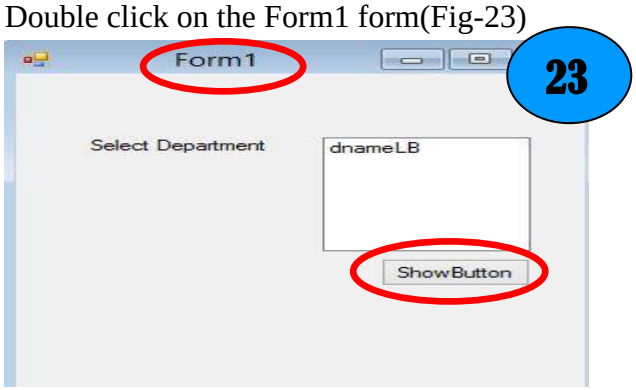
The ListBox control enables you to display a list of items to the user that the user can select by clicking.

Steps:

1. Create a desktop application and change it's properties(As like as Exercise-1)
2. Design the user interface like the following figure(Fig-22)



3. Double click on the Form1 form(Fig-23)



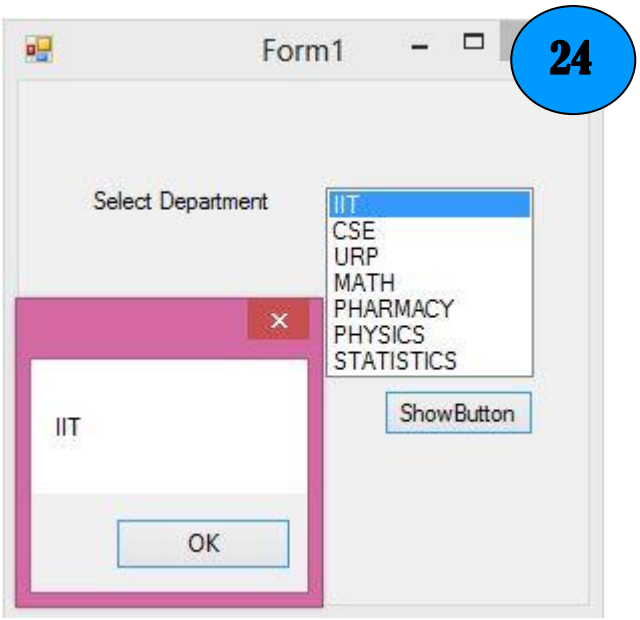
4. Write down the code snippet in the block

```
private void Form1_Load(object sender, EventArgs e)
{
    dnameLB.Items.Add("IIT");
    dnameLB.Items.Add("CSE");
    dnameLB.Items.Add("URP");
    dnameLB.Items.Add("MATH");
    dnameLB.Items.Add("PHARMACY");
    dnameLB.Items.Add("PHYSICS");
    dnameLB.Items.Add("STATISTICS");
    dnameLB.SelectionMode = SelectionMode.MultiSimple;
  }
```

5. Double click on the 'ShowButton'(Fig-23)
6. Write down the code snippet in the block

```
private void show_button_Click(object sender, EventArgs e)
{
    foreach (Object obj in dnameLB.SelectedItems)
    {
        MessageBox.Show(obj.ToString());
    }

}
```

7. Build the solution and run it and You will get the following picture(Fig-24)

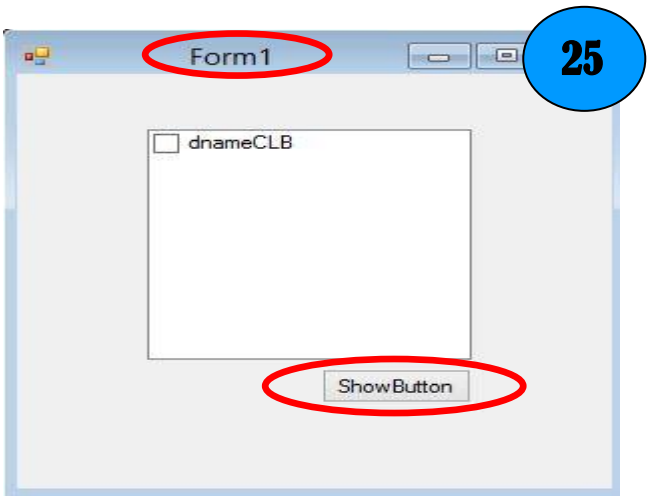


Exercise-4: Use of CheckedListBox Control

The CheckedListBox control gives you all the capability of a list box and also allows you to display a check mark next to the items in the list box.

Steps:

1. Create a desktop application and change it's properties(As like as Exercise-1)
2. Design the user interface like the following figure(Fig-25)



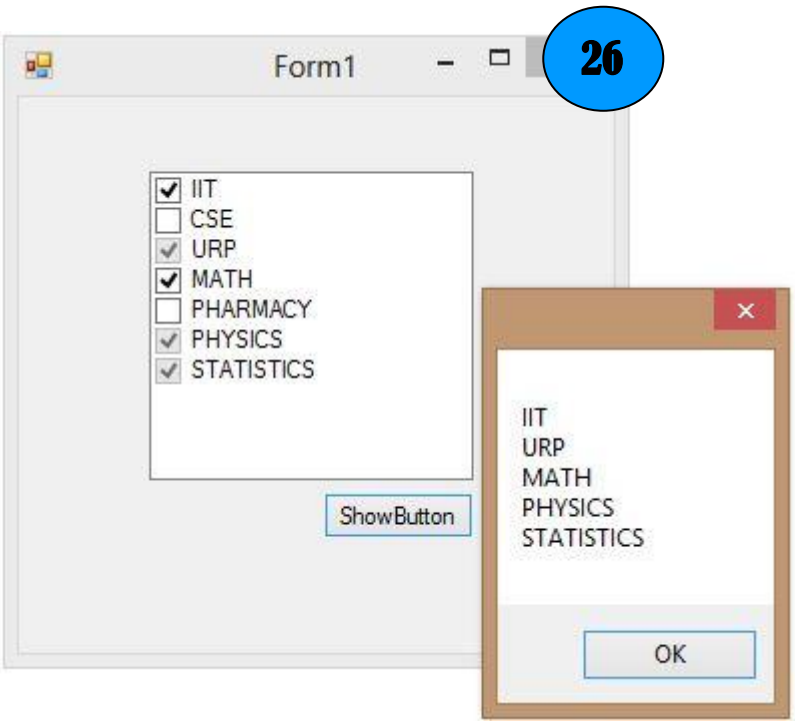
3. Double click on the Form1 form(Fig-25)
4. Write down the code snippet

```
private void Form1_Load(object sender, EventArgs e)
{
    dnameCLB.Items.Add("IIT", CheckState.Checked);
    dnameCLB.Items.Add("CSE", CheckState.Unchecked);
    dnameCLB.Items.Add("URP", CheckState.Indeterminate);
    dnameCLB.Items.Add("MATH",CheckState.Checked);
    dnameCLB.Items.Add("PHARMACY",CheckState.Unchecked);
    dnameCLB.Items.Add("PHYSICS", CheckState.Indeterminate);
    dnameCLB.Items.Add("STATISTICS", CheckState.Indeterminate);

}
```

5. Double click on the 'ShowButton'(Fig-25)
6. Write down the code snippet in the block

```
private void show_button_Click(object sender, EventArgs e)
{
    string dname="";
    foreach (Object obj in dnameCLB.CheckedItems)
    {
     dname = dname + obj.ToString()+"\n";
    }
     MessageBox.Show(dname);
}
```

7. Build the solution and run it and You will get the following picture(Fig-26)

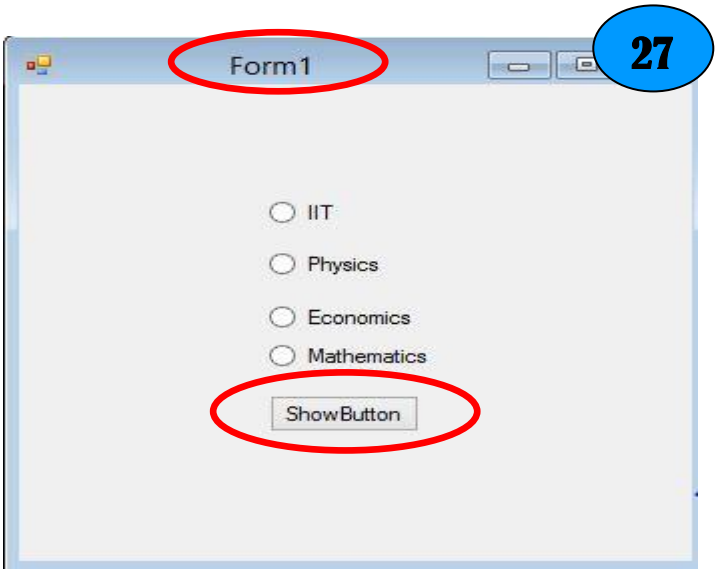


Exercise-5: Use of RadioButton Control

A RadioButton or OptionButton enables the user to select a single option from a group of choices when paired with other RadioButton controls. When a user clicks on a RadioButton, it becomes checked, and all other RadioButtons with same group become unchecked.

Steps:

1. Create a desktop application and change it's properties(As like as Exercise-1)
2. Design the user interface like the following figure(Fig-27)



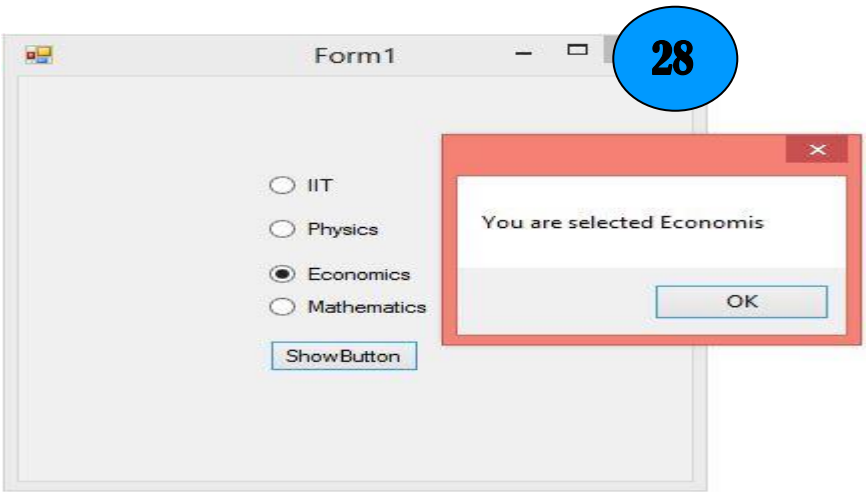
3. Double click on the Form1 form(Fig-27)
4. Write down the code snippet

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

  iitRB.Checked = true;// initial only iit checke

 }
```

5. Double click on the ShowButton(Fig-27)
6. Write down the code snippet in the block

```
private void show_button_Click(object sender, EventArgs e)
   {
    if (iitRB.Checked == true)
         {
          MessageBox.Show("You are selected IIT");
          return;
         }
     else if (phyRB.Checked == true)
       {
          MessageBox.Show("You are selected Physics ");
          return;
      }
      else if (ecoRB.Checked == true)
      {
          MessageBox.Show("You are selected Economis ");
          return;
      }
     else
      {
        MessageBox.Show("You are selected Mathematics ");
              return;
       }

   }
```

7. Build the solution and run it and You will get the following picture(Fig-28)



Exercise-6: Use of CheckBox Control

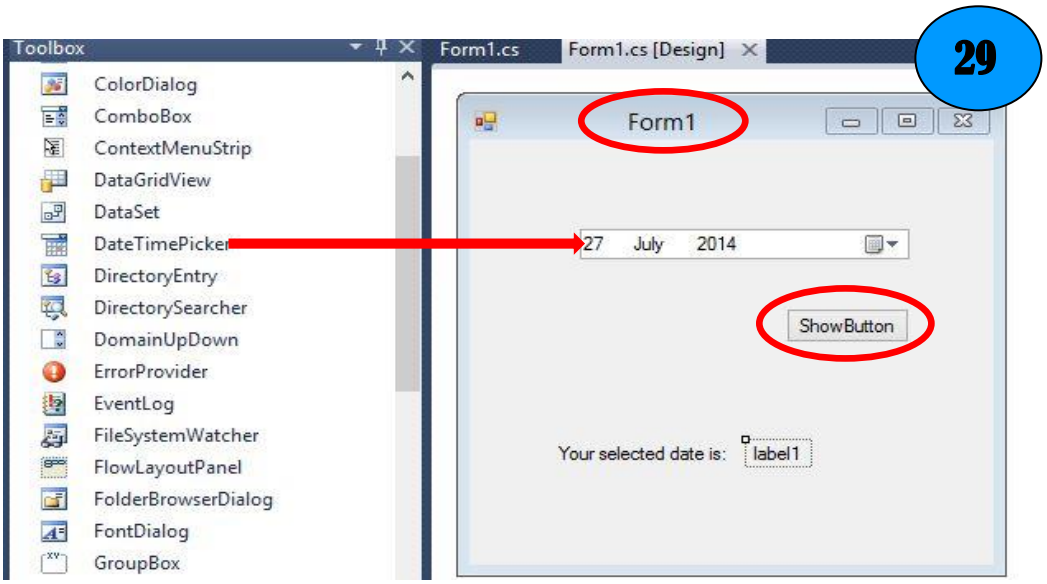
CheckBoxes allow the user to make multiple selections from a number of options. CheckBox to give the user an option, such as true/false or yes/no. You can click a check box to select it and click it again to deselect it.

Practice-1: Create a desktop application by using CheckBox control.(As like as RadioButton Control) and How to get multiple value using CheckBox.

Exercise-7: Use of DateTimePicker Control

The DateTimePicker control allows you to display and collect date and time from the user with a specified format. The DateTimePicker control has two parts, a label that displays the selected date and a popup calendar that allows users to select a new date. In this tutorial, we will see how to create a DateTimePicker control at design-time as well as at run-time, set its properties and call its methods.

1. Create a desktop application(As like as Exercise-1)
2. Change the Name Property of DateTimePicker(must be select) into dateTP (According to your choice),Change the Name property of Label into dateLB and Change the Name and Text property of Button into show_button ,ShowButton and Every time press Enter from your keyboard.(As like as Exercise-1)
3. Design the user interface like the following figure(Fig-29)



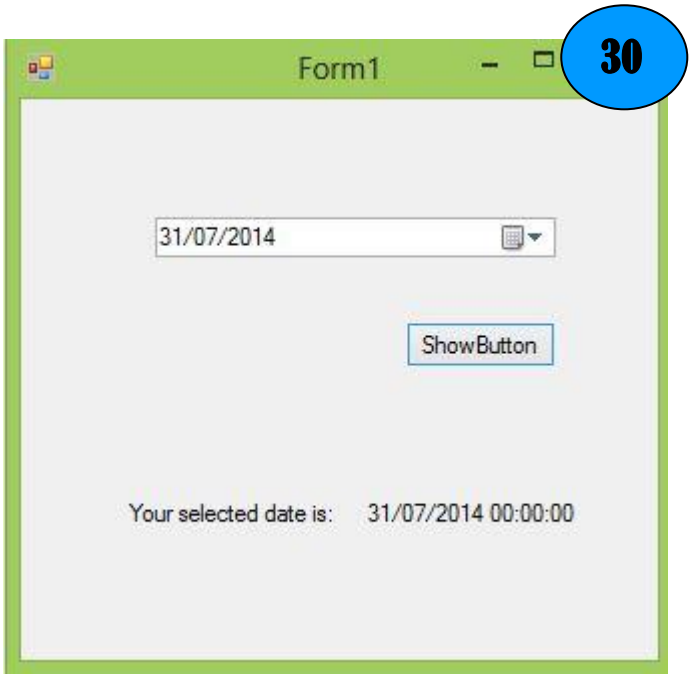
4. Double click on the Form1 form(Fig-29)
5. Write down the code snippet

```
private void Form1_Load(object sender, EventArgs e)
{
    dateTP.Format = DateTimePickerFormat.Short;
    dateTP.Value = DateTime.Today;
}
```

6. Double click on the ShowButton(Fig-29)
7. Write down the code snippet in the block

```
private void show_button_Click(object sender, EventArgs e)
 {
    DateTime vdate = dateTP.Value;
    dateLB.Text = vdate.ToString(); //Show output in label

 }
```

8. Build the solution and run it and here you will see output in the label and You will get the following picture (Fig-30)

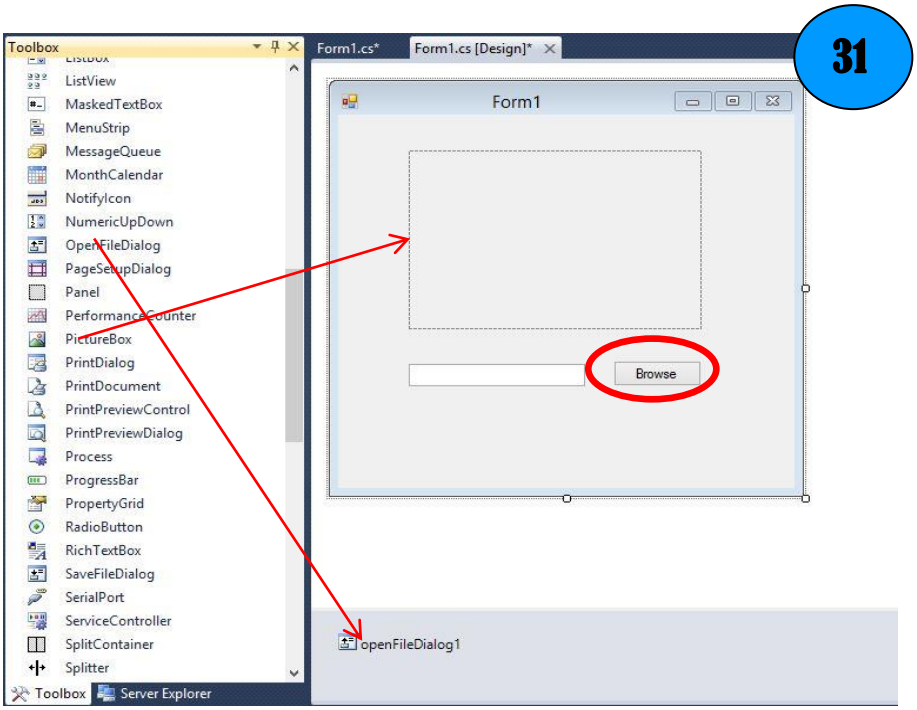


Exercise-8: Use of OpenFileDialog and PictureBox Control

OpenFileDialog allows users to browse folders and select files. It is available in Windows Forms and can be used with C# code. It displays the standard Windows dialog box. The results of the selection can be read in your C# code.The Windows Forms PictureBox control is used to display images in bitmap, GIF , icon , or JPEG formats.

Steps:

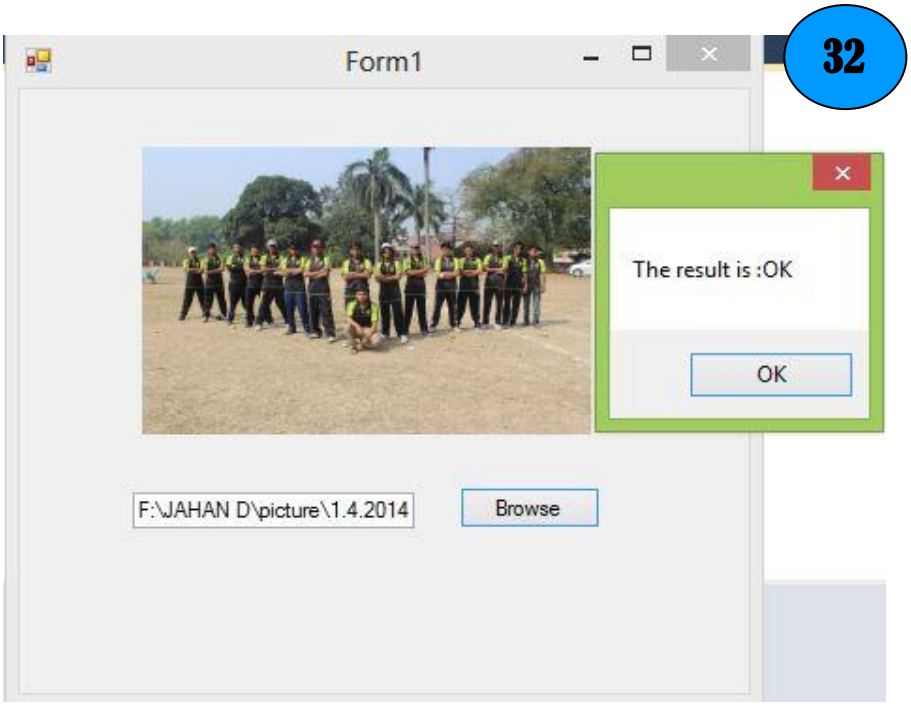
1. Create a desktop application (As like as Exercise-1)
2. Change the Name Property of PictureBox(must be select) into pictureBox and Change the Name and Text property of Button into browse_button ,Browse and Change the Name property of TextBox into filenameTB and openFileDialog1 is No need to change of OpenFileDialog Toolbox and Every time press Enter from your keyboard.(As like as Exercise-1)
3. Design the user interface like the following figure(Fig-31)



4. Double click on the Browse Button(Fig-31)
5. Write down the code snippet in the block

```
private void browse_button_Click(object sender, EventArgs e)
      {
 // Show the dialog and get result.
 DialogResult result = openFileDialog1.ShowDialog();
 openFileDialog1.Filter = "Image Files(*.jpg; *.jpeg; *.gif; *.bmp)|*.jpg;
 *.jpeg; *.gif; *.bmp";
 if (result == DialogResult.OK) // Test result.
      {
      pictureBox.Image = new Bitmap(openFileDialog1.FileName);
      pictureBox.SizeMode = PictureBoxSizeMode.Zoom;
      // image file path
      filenameTB.Text = openFileDialog1.FileName;
      }
      MessageBox.Show("The result is :" + result);

      }
```

6. Build the solution and run it and You will get the following picture after browse(Fig-32)

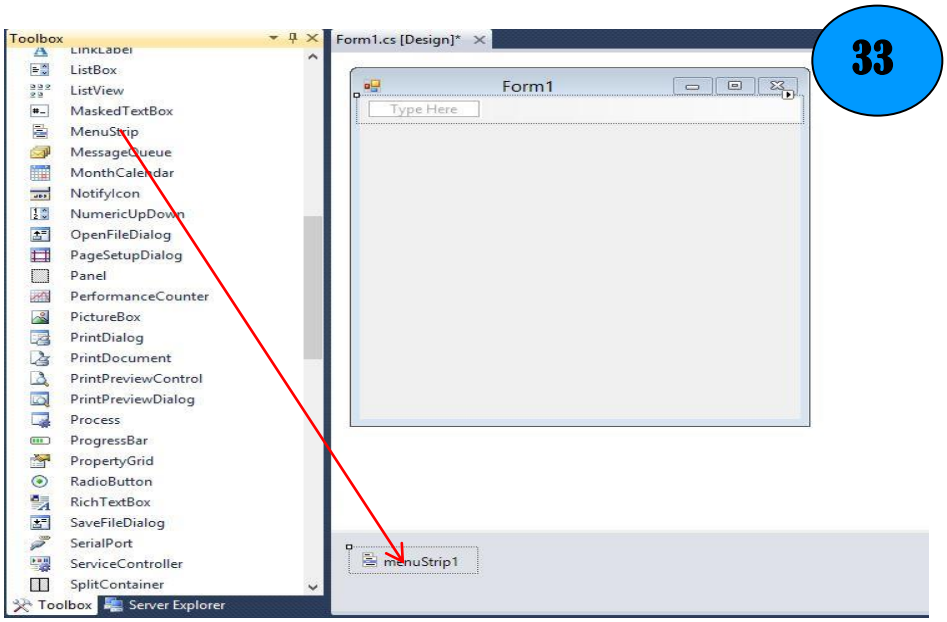


Exercise-9: Use of Menu Control

MenuStrip adds a menu bar to your Windows Forms program. With this control, we add a menu area and then add the default menus or create custom menus directly in Visual Studio.

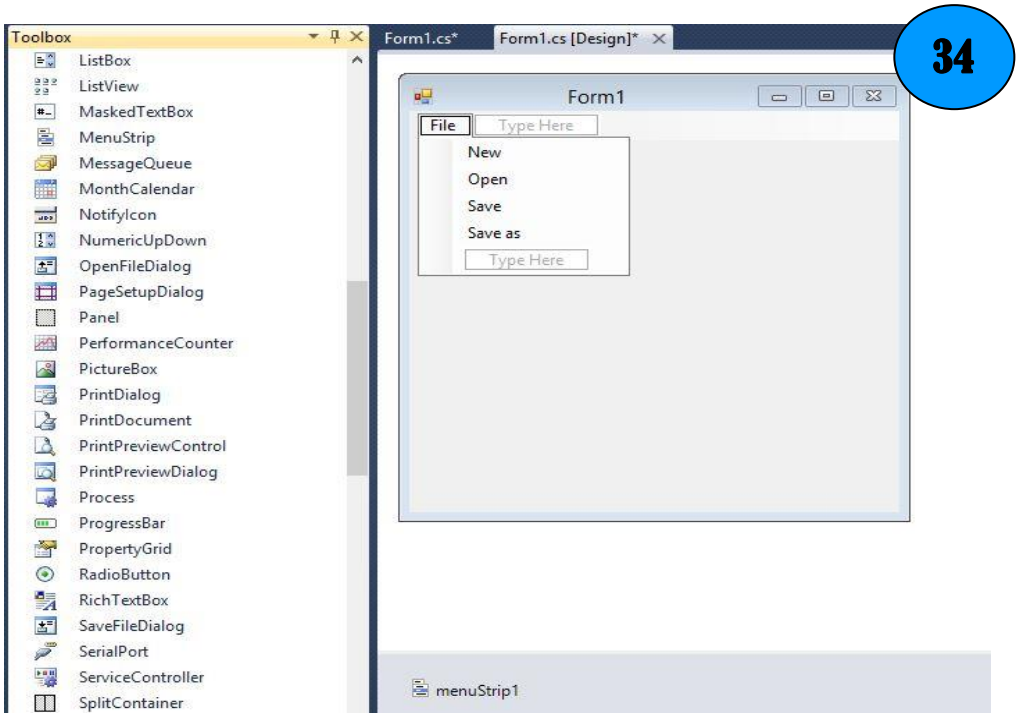
Steps:

1. Create a desktop application
2. From the Toolbox, drag a MenuStrip control onto the form(Fig-33).

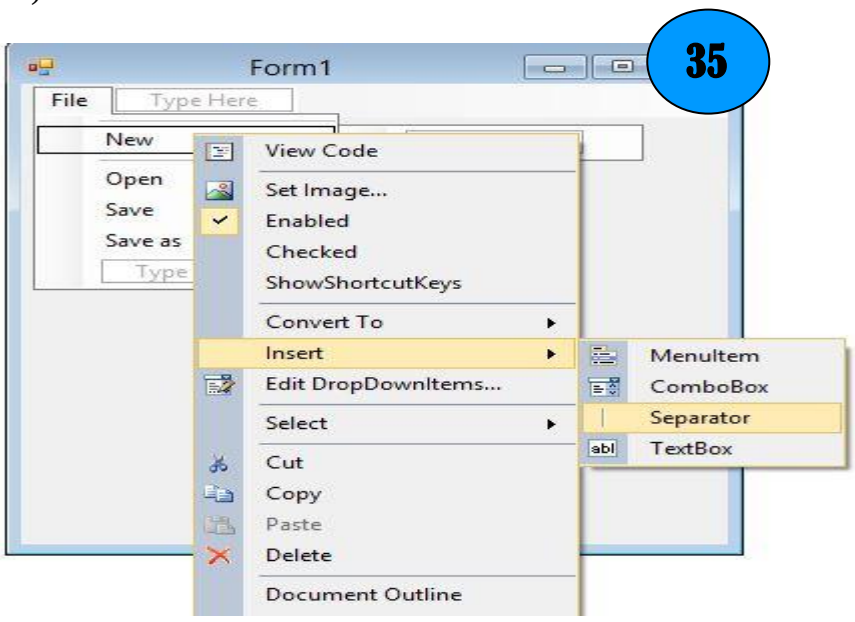


Prepared by:

3. After drag the Menustrip on your form you can directly create the menu items by type a value into the "Type Here" box(Fig-33) on the menubar part of your form. From the following picture you can understand how to create each menu items on mainmenu Object(Fig-34).

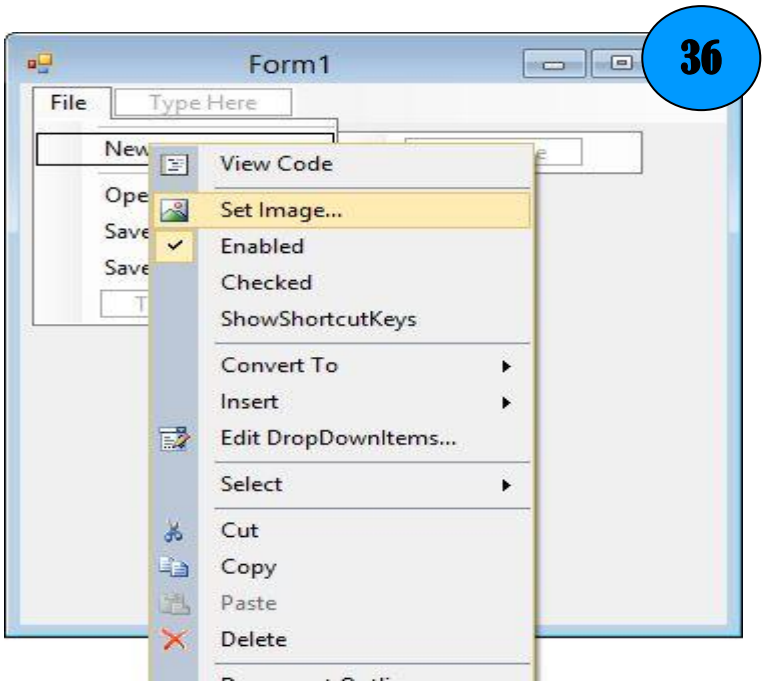


4. If you need a seperator bar , right click on your menu then go to insert->Seperator(Fig-35).

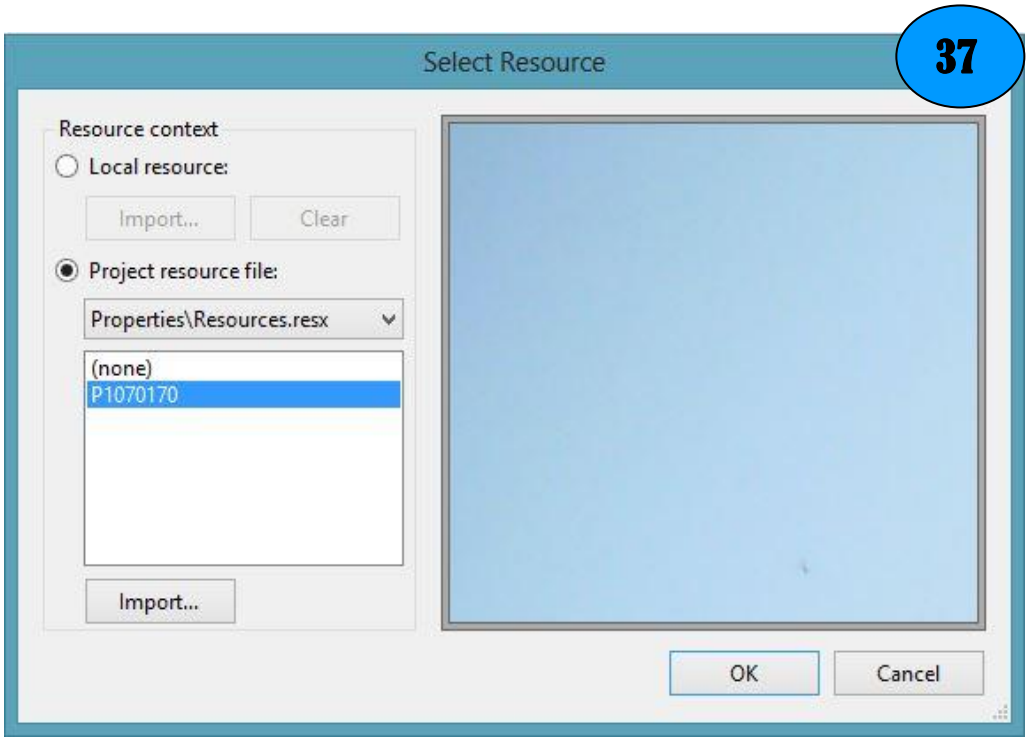


Prepared by:

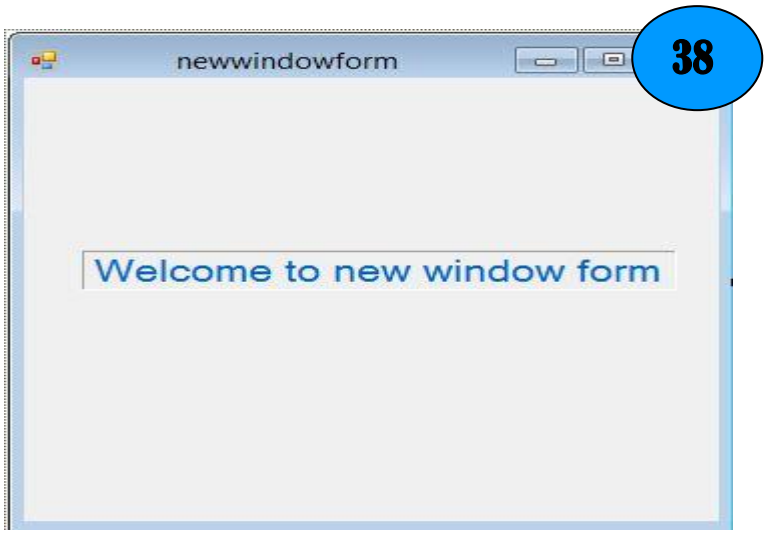
5. If you need a icon beside the menu item , right click on your menu then go to Set Image(Fig-36).



6. After clicking Set Image you will get the following picture. If there has no image then can import image.If there has existing image then you select this image and click OK(Fig-37).



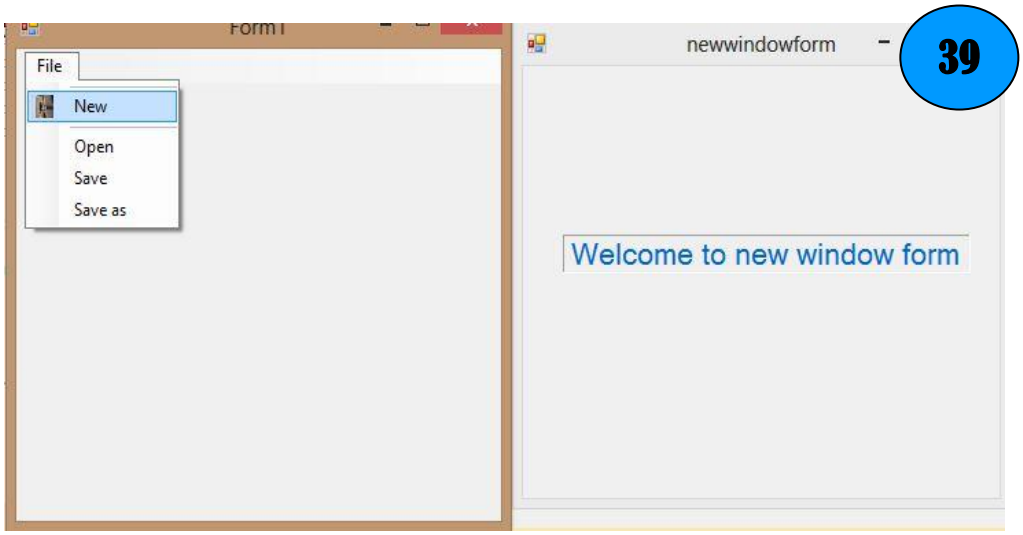
7. After creating the Menu on the form , you have to double click on each menu item and write the programs there depends on your requirements. Here we include a new window form.When we click New menu item then we will go to new window form.(See Task-2 ,How to add a new item or new window form) .(Fig-38)



8. The following C# program shows how to show a new window form when clicking a Menu item.

```
private void newToolStripMenuItem_Click(object sender, EventArgs e)
{
    newwindowform obj = new newwindowform();
    obj.Show();
    //this.Hide();
}
```

9. Build the solution and run it and You will get the following picture(Fig-39)

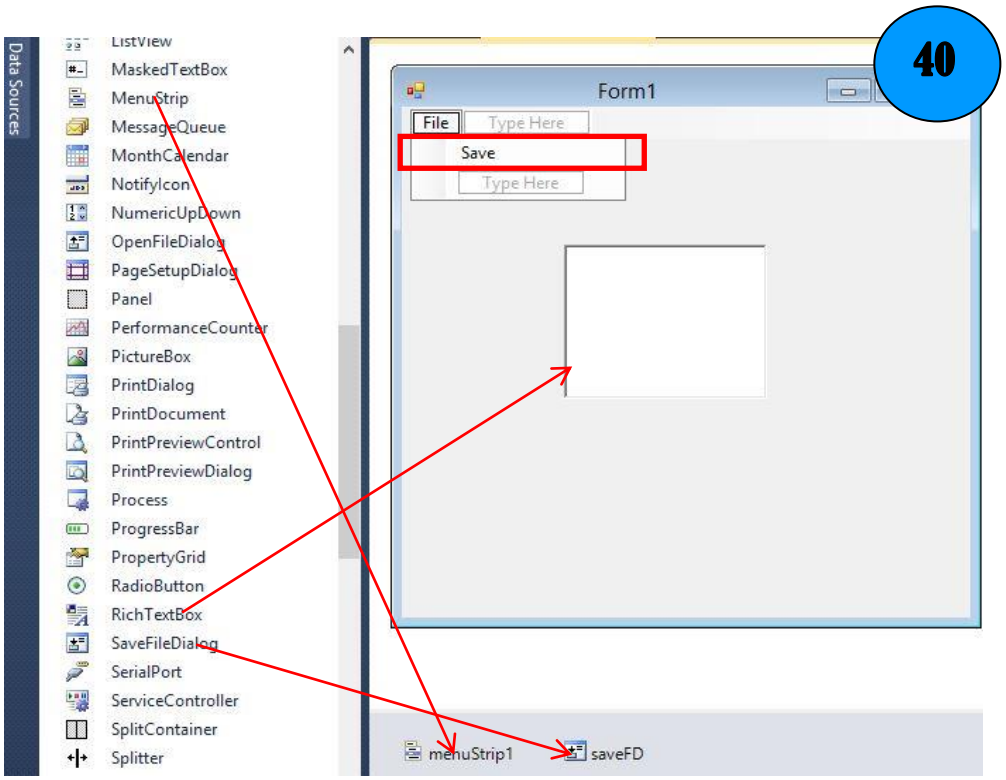


Exercise-10: Use of SaveFileDialog Control

SaveFileDialog displays a dialog box that prompts the user to select a location for file saving.

Steps:

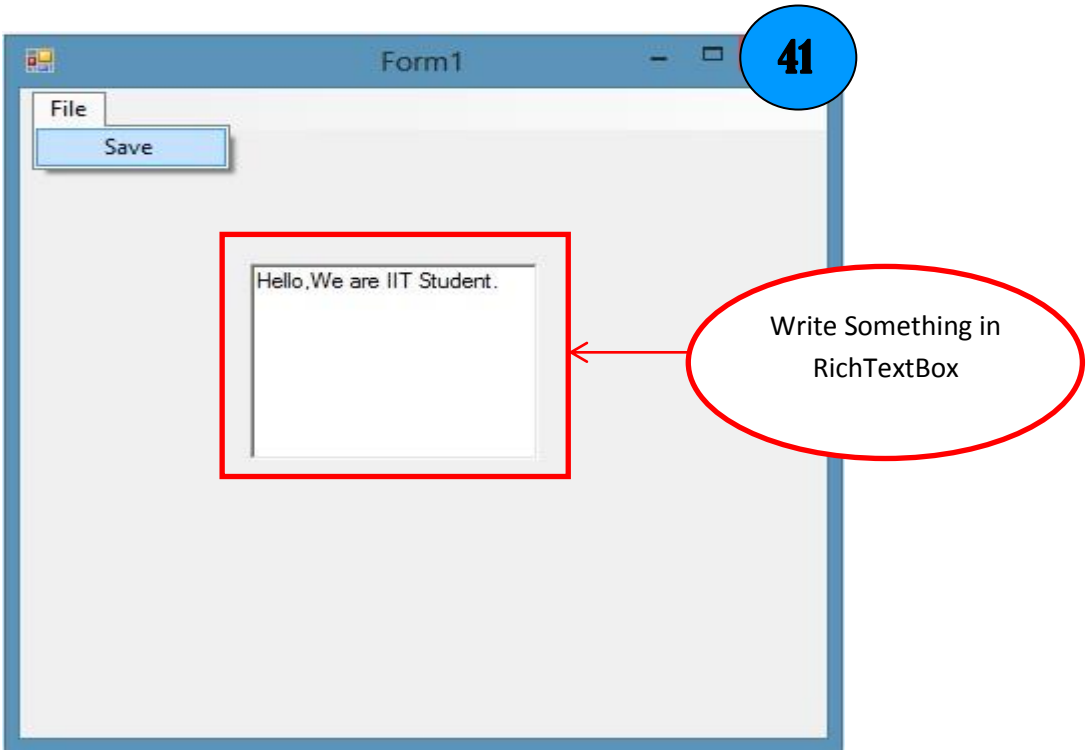
1. Create a desktop application
2. From the Toolbox, drag a MenuStrip,SaveFileDialog and RichTextBox control onto the form.After drag the Menustrip on your form you can directly create the menu items by type a value into the "Type Here" box on the menubar part of your form.Change the properties of MenuStrip,SaveFileDiallog and RichTextBox(Fig-40)



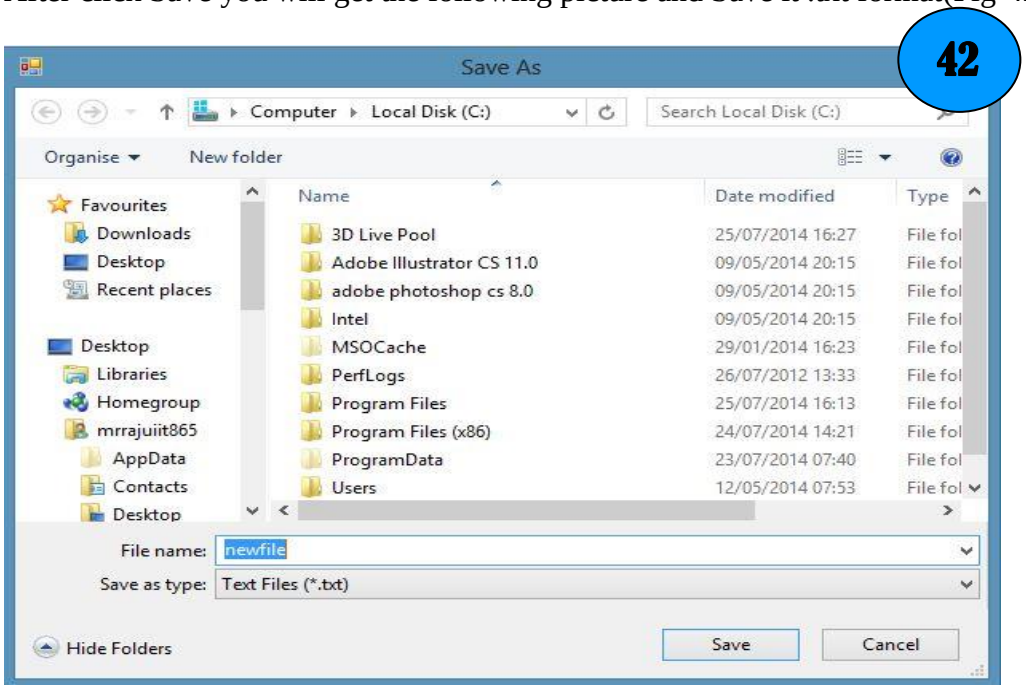
3. Double click on the Save Menu Item(Fig-40).
4. Write down the code snippet in the block

```
private void saveToolStripMenuItem_Click(object sender, EventArgs e)
      {
        string save_file = "";
        saveFD.InitialDirectory = "C:";
        saveFD.Title = "";
        saveFD.FileName = "";
        saveFD.Filter = "Text Files (*.txt)|*.txt|All Files (*.*)|*.*";
        if (saveFD.ShowDialog() !=DialogResult.Cancel)
        {
            save_file = saveFD.FileName;
            richTextBox1.SaveFile(save_file, RichTextBoxStreamType.PlainText);
       }
      }
```

5. Build the solution and run .You will get the following picture (Fig-41)



6. After click Save you will get the following picture and Save it .txt format(Fig-42).

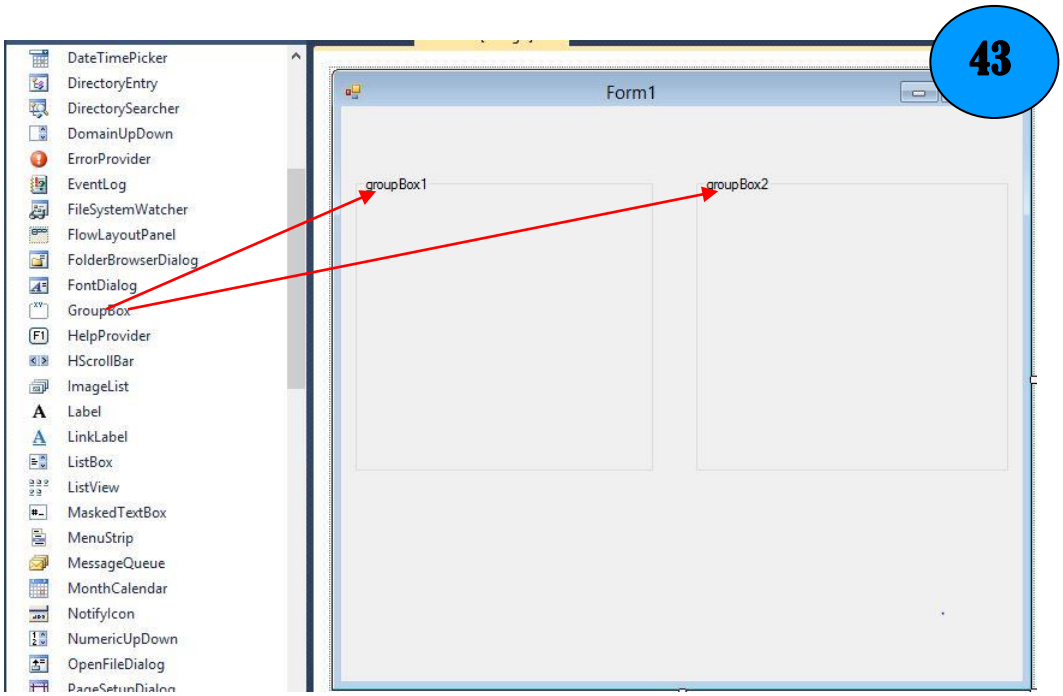


Exercise 11: Use of GroupBox Control

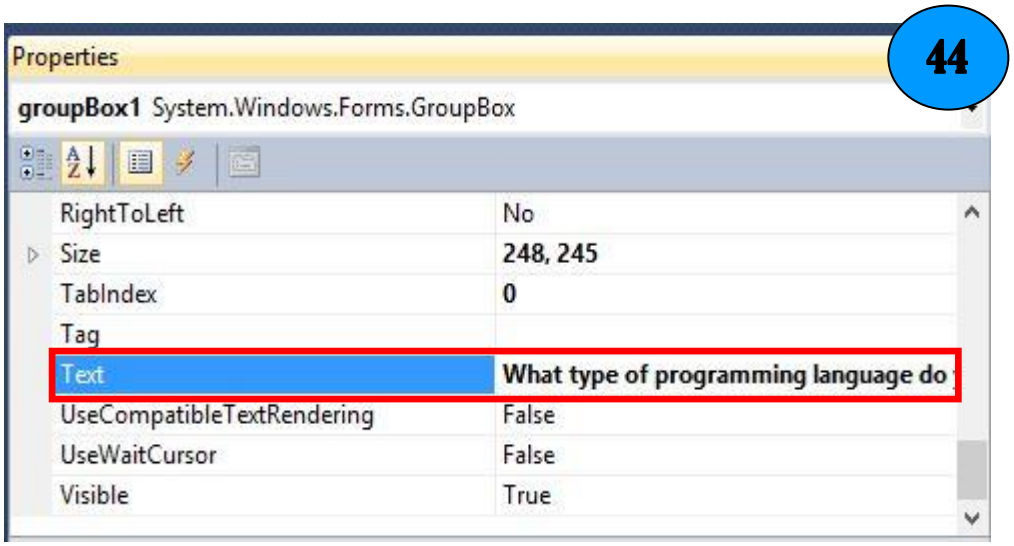
GropsBox dispalays a frame around a group of controls within an optional caption.

Steps:

1. Create a desktop application
2. From the Toolbox, drag two GroupBox on the form(Fig-43)

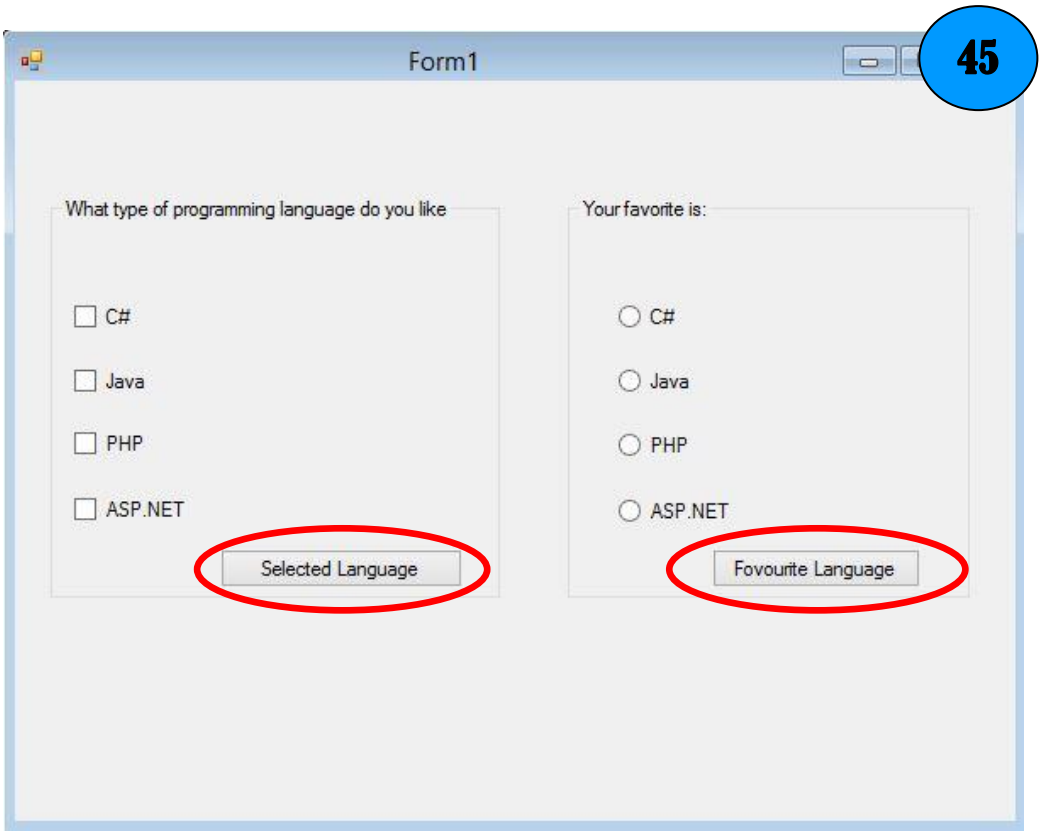


3. Select GroupBox Toolbox and rightclick on the groupbox11 and go to the Properties and change the Text property of the GroupBox control to the following text: What type of programming language do you like (Fig-44)



Prepared by:

4. After change the GroupBox properties then design the user interface with GroupBox like the following figure(Fig-45)



7. Double click on Selected Language Button(Fig45-) and write down the code snippet in the block

```
private void selected_button_Click(object sender, EventArgs e)
    {
        string language = "";
        if (checkBox1.Checked)
        {
            language = language + checkBox1.Text + "\n";
        }
        if (checkBox2.Checked)
        {
            language = language + checkBox2.Text + "\n";
        }
        if (checkBox3.Checked)
        {
            language = language + checkBox3.Text + "\n";
        }
        if (checkBox4.Checked)
        {
            language = language + checkBox4.Text + "\n";
        }
        MessageBox.Show(language);

    }
```

8. Double click on the Favourite Language(Fig-45) and write down the code snippet in the block

```
private void favourite_button_Click(object sender, EventArgs e)
    {
        string favolanguage = "";
        if (radioButton1.Checked)
        {
            favolanguage = radioButton1.Text;
        }
        else  if (radioButton2.Checked)
        {
            favolanguage = radioButton2.Text;
        }
        else if (radioButton3.Checked)
        {
            favolanguage = radioButton3.Text;
        }
        else
        {
            favolanguage = radioButton4.Text;
        }
        MessageBox.Show("Your favourite language is:"+ favolanguage);
    }
```

9. Build the solution and run .You will get the following picture (Fig-46)