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

<u>Objectives</u>

After completing this module you will be able to:

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

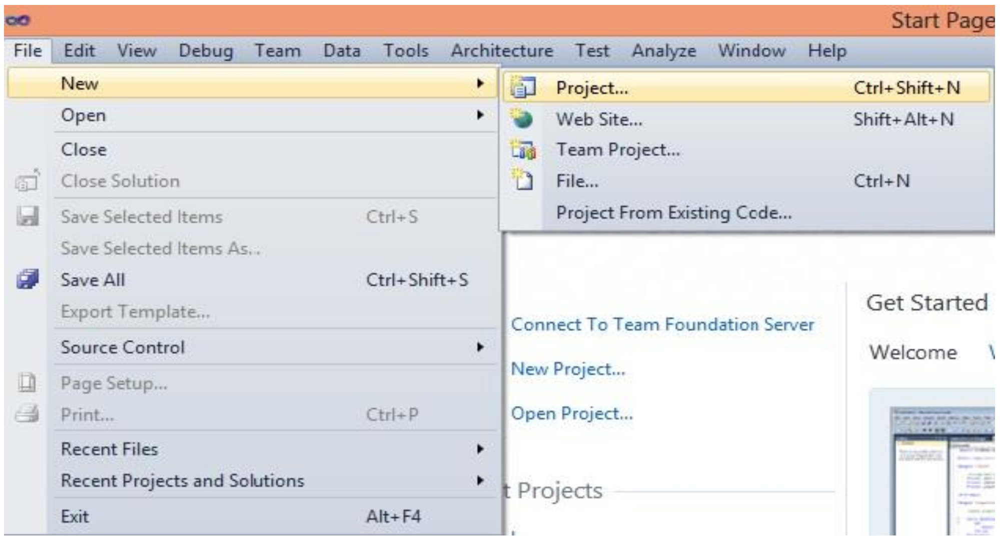
Task 1: Installation process of Visual Studio

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

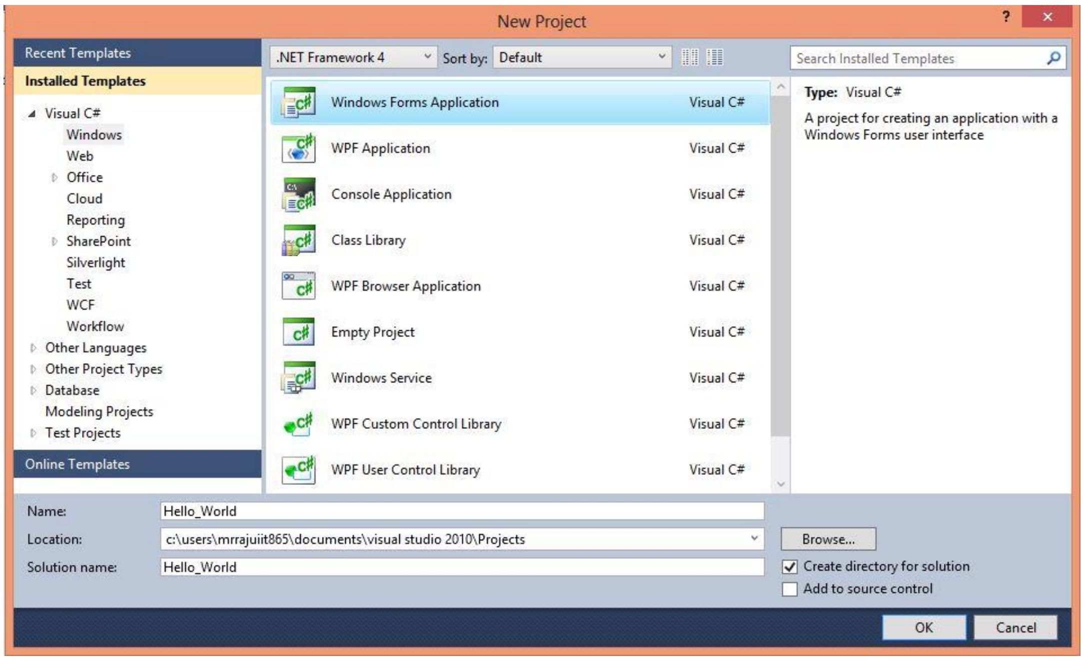
Now we will create a very simple first desktop application. In this application, user will click on a button 'Show' and a messagebox (a small popup box) will appear displaying 'Hello World'.

Steps:

1. Start Microsoft Visual Studio 2010
2. Click File>New>Project



3. Select Windows>Windows Forms Application
4. Give an appropriate name and others
5. Click 'Ok' button



Type: Visual C#

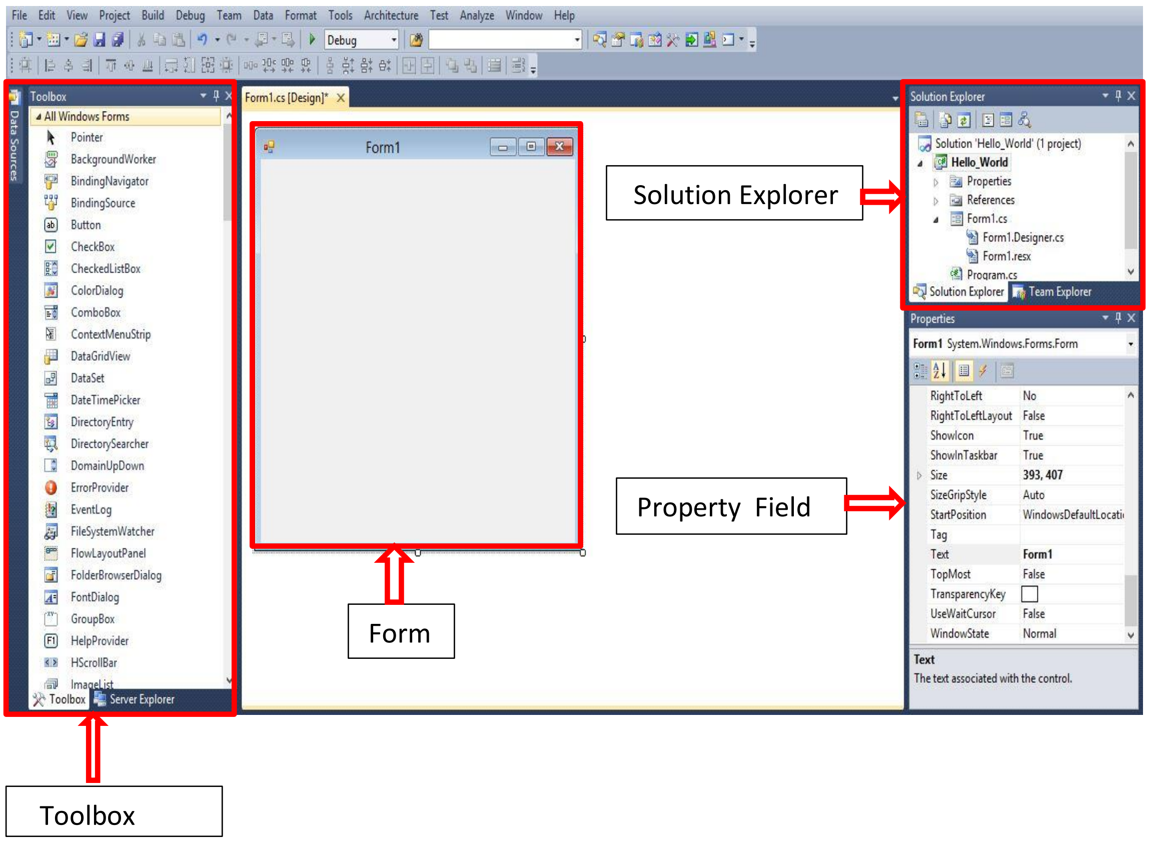
A project for creating an application with a Windows Forms user interface

Name:	Hello_World
Location:	c:\users\mrrajuiit865\documents\visual studio 2010\Projects
Solution name:	Hello_World

☑ Create directory for solution

☐ Add to source control

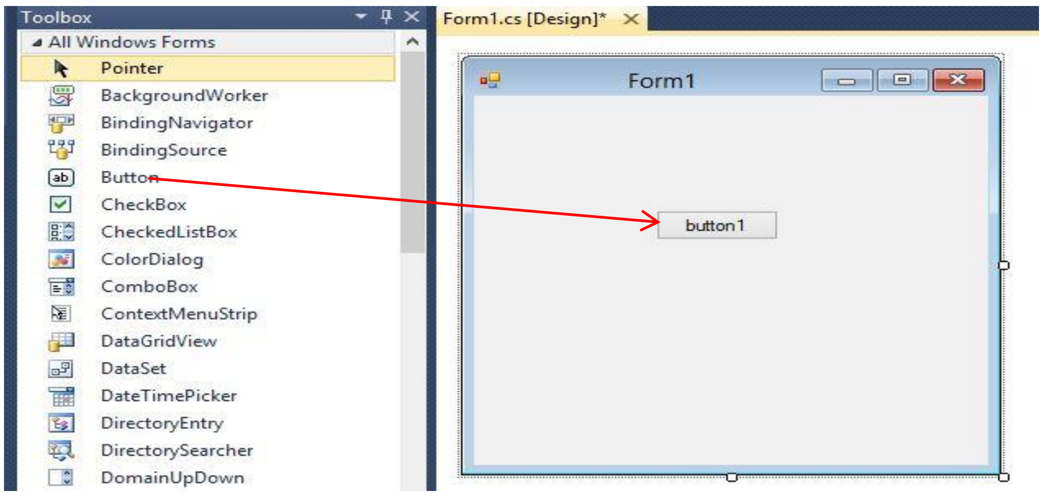
You will get the following picture in your window



RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	393, 407
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocati
Tag	
Text	Form1
TopMost	False
TransparencyKey	☐
UseWaitCursor	False
WindowState	Normal

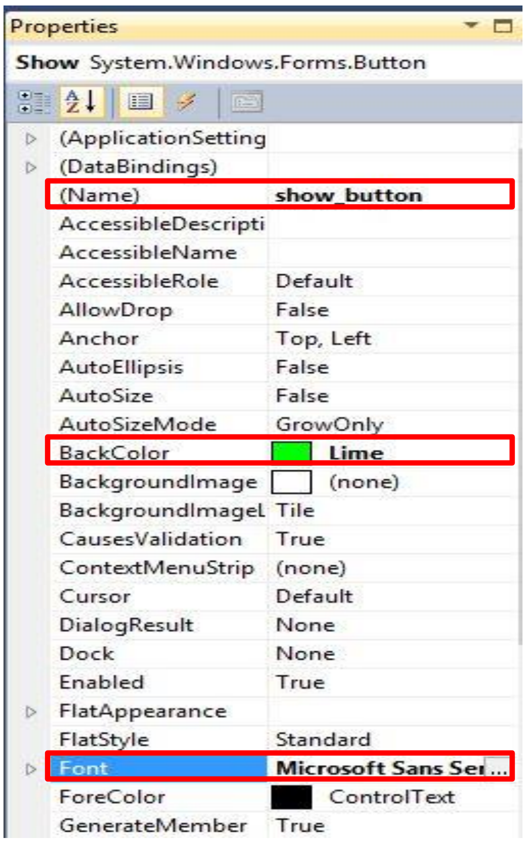
Text
The text associated with the control.

6. Now drag and drop a Button from Toolbox situated at the left side of your development window and you will get the following picture



7. Click on the Button and Change the Name Property,Backcolor,Front and Text Property of the Button(must be select) 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.

This will looks like the following picture


Properties

Show System.Windows.Forms.Button

(ApplicationSetting	
(DataBindings)	
(Name)	**show_button**
AccessibleDescripti	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left
AutoEllipsis	False
AutoSize	False
AutoSizeMode	GrowOnly
BackColor	🟩 **Lime**
BackgroundImage	☐ (none)
BackgroundImageL	Tile
CausesValidation	True
ContextMenuStrip	(none)
Cursor	Default
DialogResult	None
Dock	None
Enabled	True
FlatAppearance	
FlatStyle	Standard
Font	**Microsoft Sans Ser** ...
ForeColor	■ ControlText
GenerateMember	True

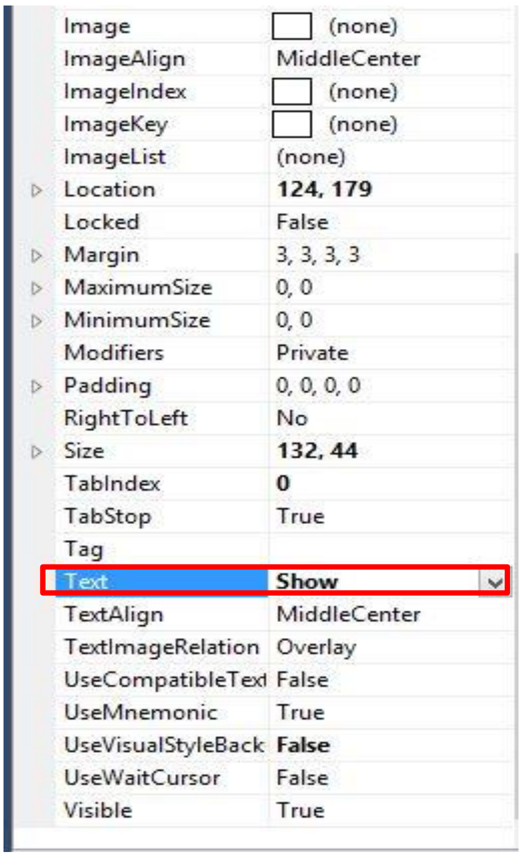
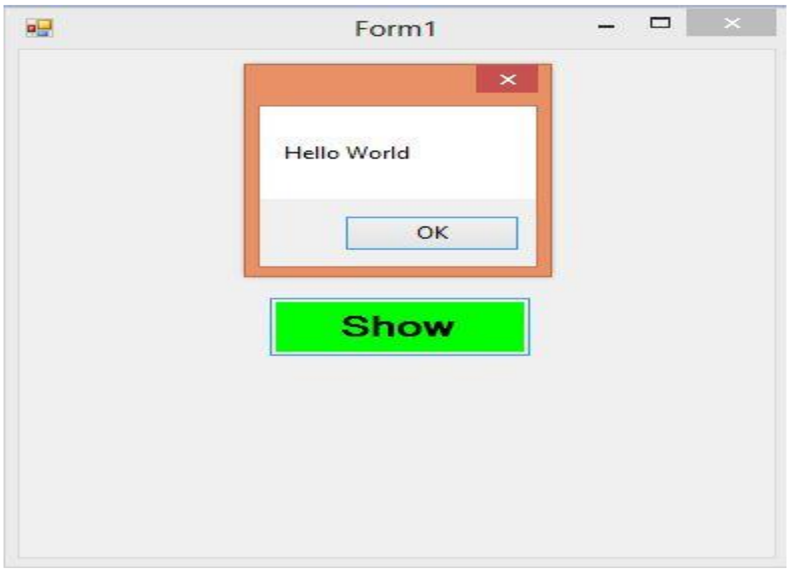


Image	☐ (none)
ImageAlign	MiddleCenter
ImageIndex	☐ (none)
ImageKey	☐ (none)
ImageList	(none)
Location	**124, 179**
Locked	False
Margin	3, 3, 3, 3
MaximumSize	0, 0
MinimumSize	0, 0
Modifiers	Private
Padding	0, 0, 0, 0
RightToLeft	No
Size	**132, 44**
TabIndex	**0**
TabStop	True
Tag	
Text	**Show**
TextAlign	MiddleCenter
TextImageRelation	Overlay
UseCompatibleText	False
UseMnemonic	True
UseVisualStyleBack	**False**
UseWaitCursor	False
Visible	True

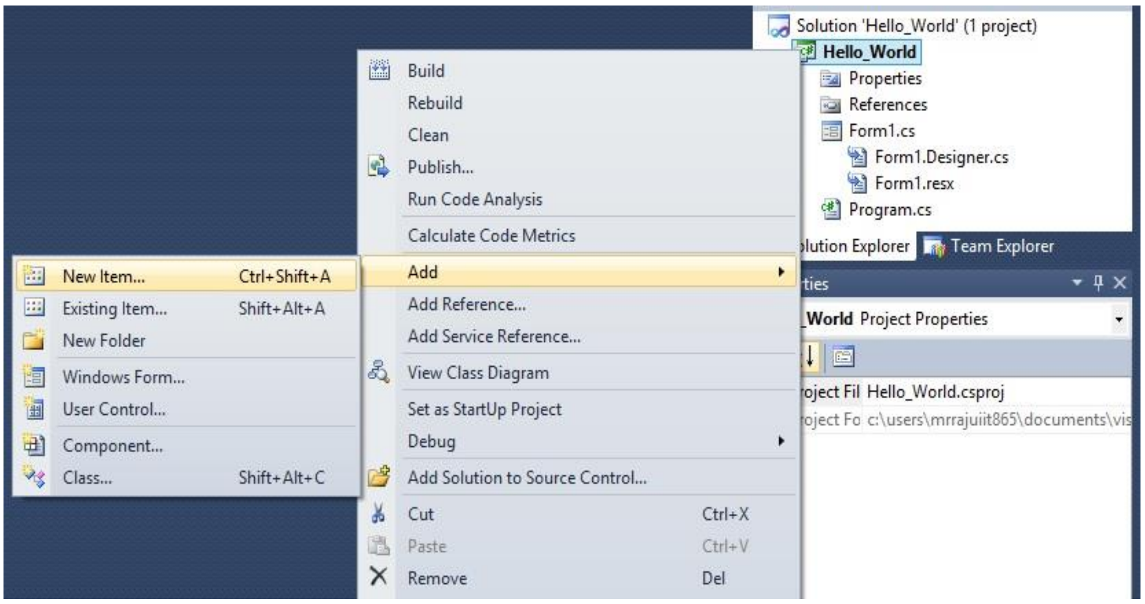
8. Double click on the Show Button of the form and write the following code snippet

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

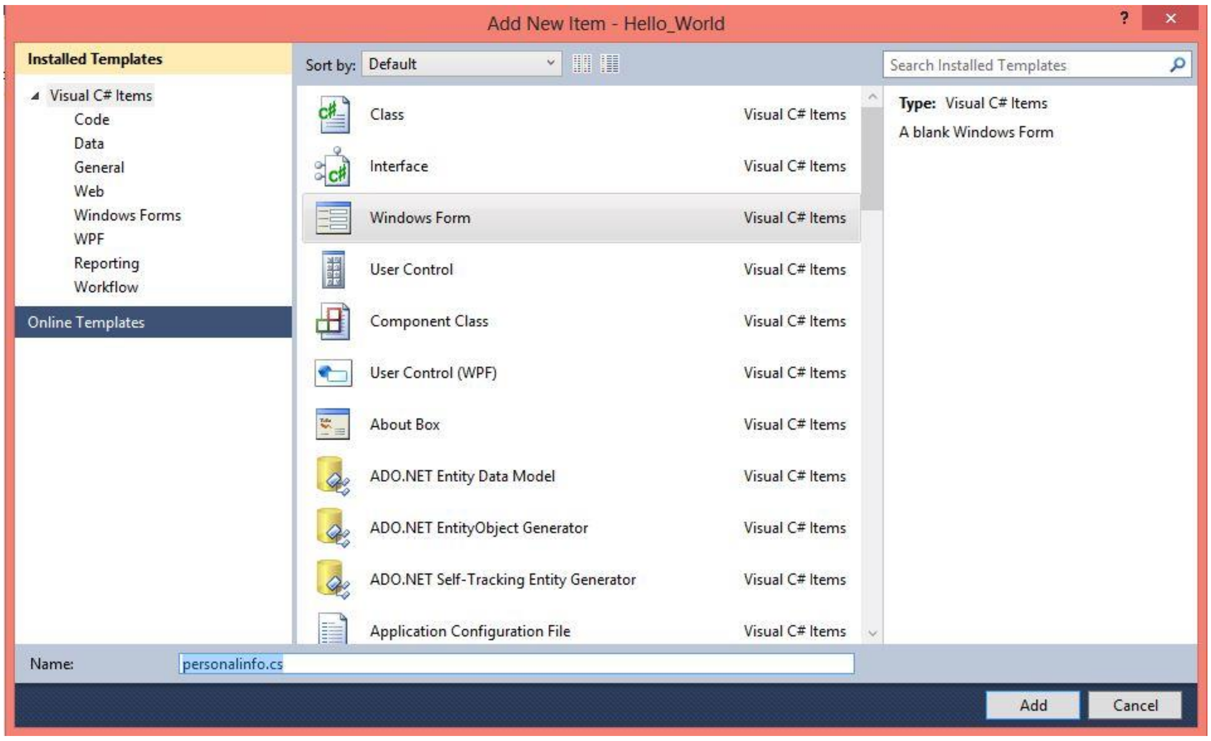
9. Now run the application by clicking on the debug button or pressing F5
10. Click on the ‘Show’ button of the running application
11. A messagebox will appear. You will get the following picture



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



After click the New Item You will get the following picture:



Sort by: Default

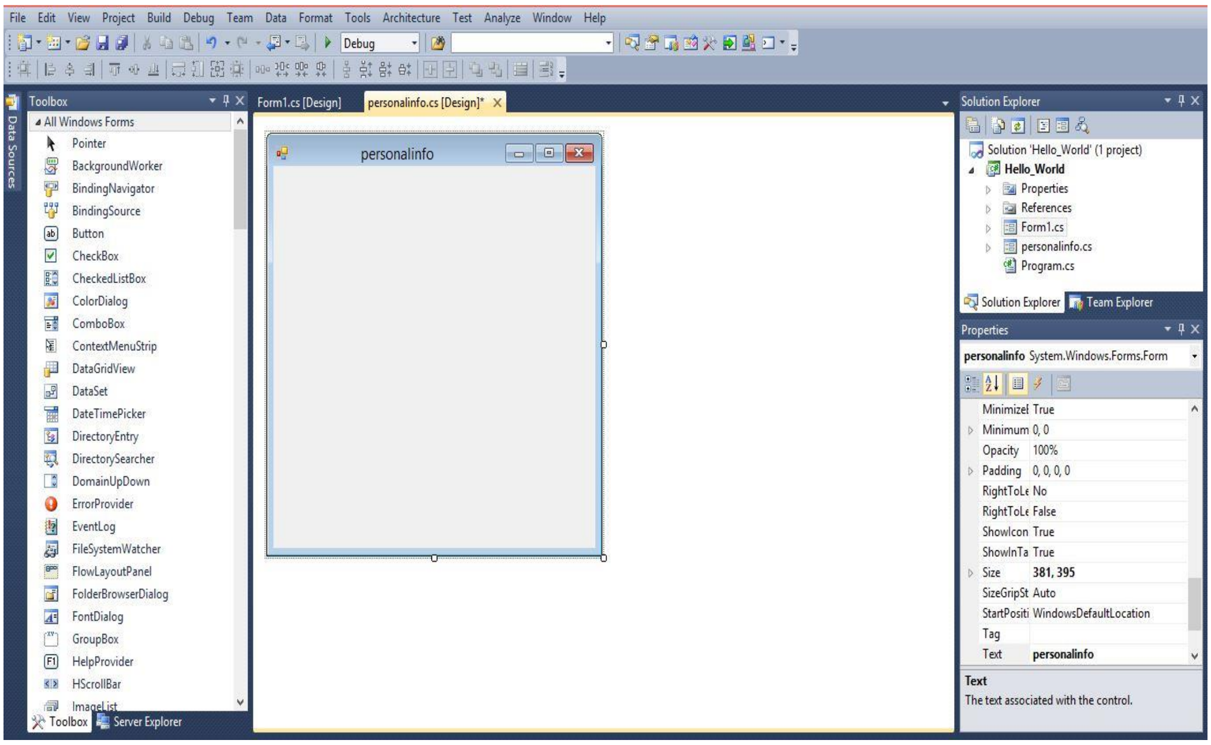
Class	Visual C# Items
Interface	Visual C# Items
Windows Form	Visual C# Items
User Control	Visual C# Items
Component Class	Visual C# Items
User Control (WPF)	Visual C# Items
About Box	Visual C# Items
ADO.NET Entity Data Model	Visual C# Items
ADO.NET EntityObject Generator	Visual C# Items
ADO.NET Self-Tracking Entity Generator	Visual C# Items
Application Configuration File	Visual C# Items

Type: Visual C# Items

A blank Windows Form

Name: personalinfo.cs

Here You select Windows Form and change the Name into personalinfo.cs .After click the Add button you will get the new windows form in the same project and done another work.



Solution 'Hello_World' (1 project)
Hello_World
Properties
References
Form1.cs
personalinfo.cs
Program.cs

Properties

personalinfo System.Windows.Forms.Form

Minimizel	True
Minimum	0, 0
Opacity	100%
Padding	0, 0, 0, 0
RightToLe	No
RightToLe	False
ShowIcon	True
ShowInTa	True
Size	381, 395
SizeGripSt	Auto
StartPositi	WindowsDefaultLocation
Tag	
Text	personalinfo

Text
The text associated with the control.

Note 1: 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;
```

If this is not written at the beginning of a class structure, one can't use the regular expression functionalities under that class.

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 3: 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.

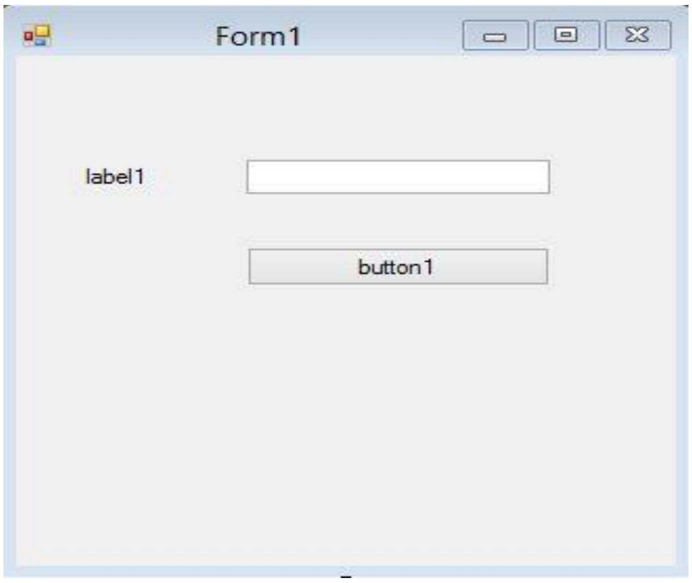
A **Button** is a control, which is an interactive component that enables users to communicate with an application. The Button class inherits directly from the ButtonBase class. A Button can be clicked by using the mouse, ENTER key, or SPACEBAR if the Button has focus.

To create a user interface with Label , Textbox and Button controls

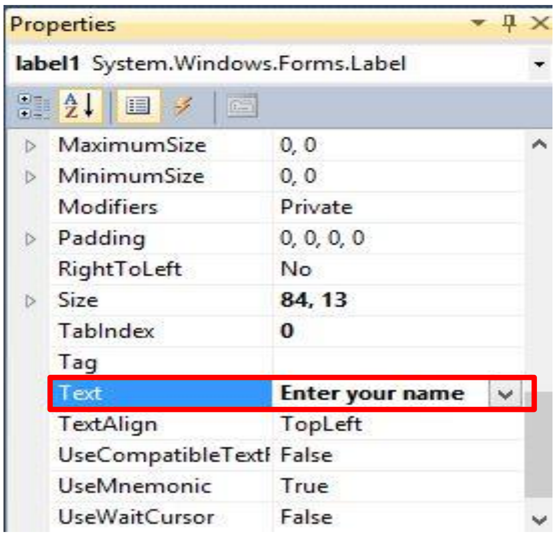
1. Start Microsoft Visual Studio 2010
2. Click File>New>Project
3. Select Windows>Windows Forms Application
4. In the Name box, type TextBoxExample, and then click OK.

   A new Windows Forms project opens.

5. From the Toolbox, drag a TextBox, Label, and Button control onto the form.



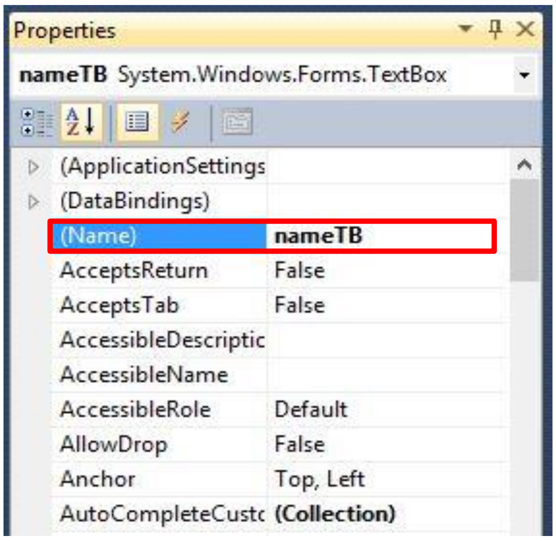
6. Select Label Toolbox and go to the Properties window, change the Text property of the Label control to the following text: Enter your name


Properties

label1 System.Windows.Forms.Label

MaximumSize	0, 0
MinimumSize	0, 0
Modifiers	Private
Padding	0, 0, 0, 0
RightToLeft	No
Size	84, 13
TabIndex	0
Tag	
Text	Enter your name
TextAlign	TopLeft
UseCompatibleTextI	False
UseMnemonic	True
UseWaitCursor	False

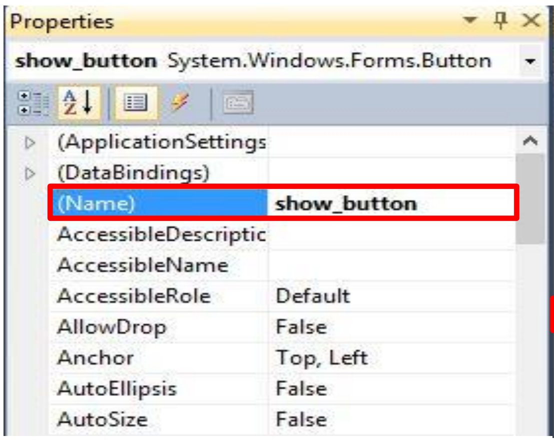
Select TextBox Toolbox and go to the Properties window, change the Name property of the TextBox control to the following text: nameTB


Properties

nameTB System.Windows.Forms.TextBox

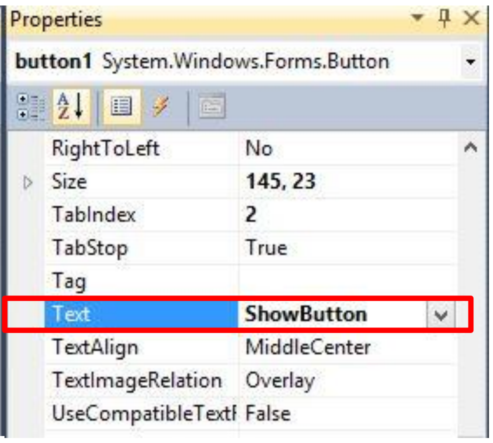
(ApplicationSettings	
(DataBindings)	
(Name)	nameTB
AcceptsReturn	False
AcceptsTab	False
AccessibleDescriptic	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left
AutoCompleteCustc	(Collection)

Select Button Toolbox 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


Properties

show_button System.Windows.Forms.Button

(ApplicationSettings	
(DataBindings)	
(Name)	show_button
AccessibleDescriptic	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
Anchor	Top, Left
AutoEllipsis	False
AutoSize	False

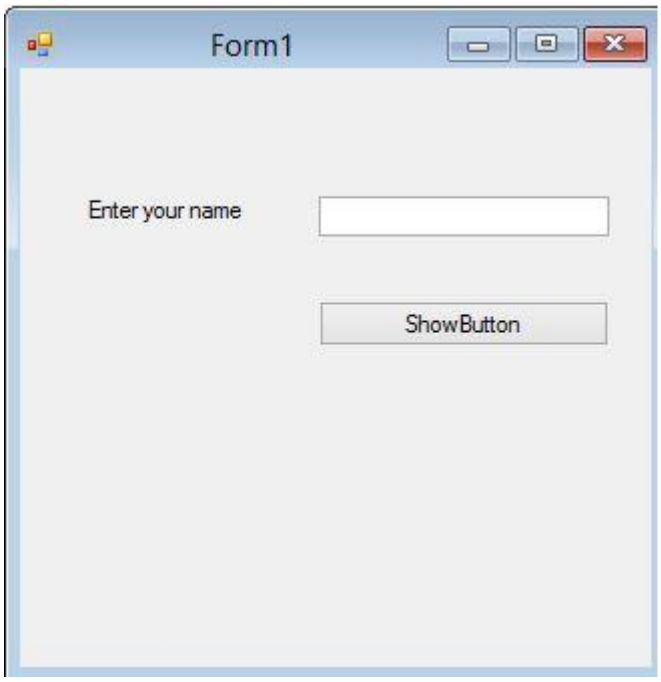

Properties

button1 System.Windows.Forms.Button

RightToLeft	No
Size	145, 23
TabIndex	2
TabStop	True
Tag	
Text	ShowButton
TextAlign	MiddleCenter
TextImageRelation	Overlay
UseCompatibleTextI	False

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

7. After doing this the windows form look like as:



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

1. Double-click the Button control to open the Code Editor.

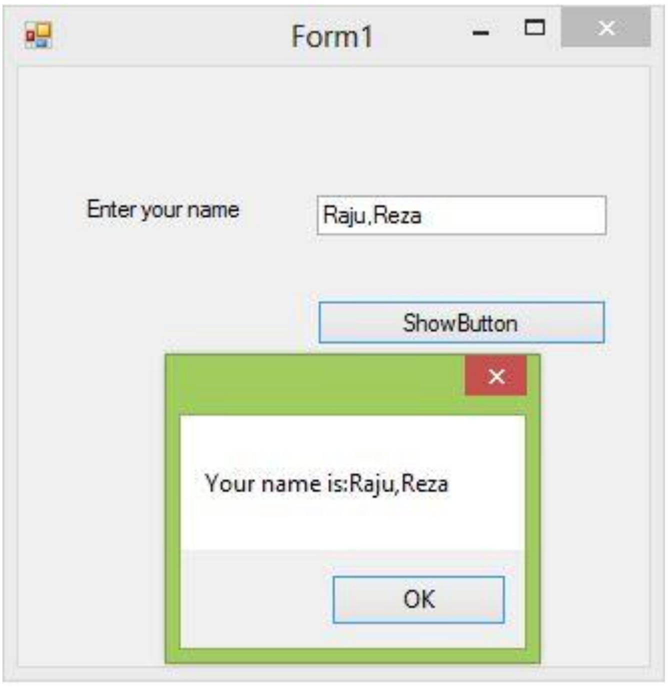
   The Code Editor opens the show_button_Click event handler.

2. 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);

    }
```

3. Press F5 to run your program.
4. When the form appears, type your name in the TextBox control and click the button. A message box appears that displays the text in the TextBox control. Change the text

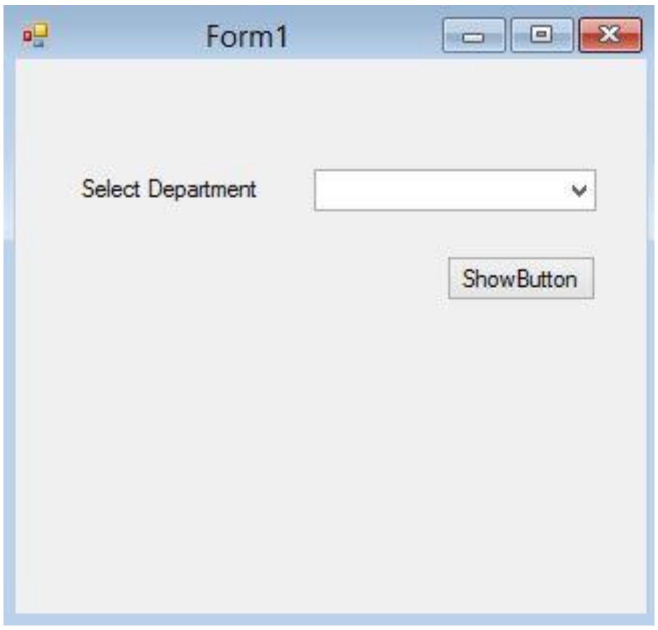
and click the button again. Each time that you click the button, the updated text is displayed.You will get the following picture:



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.

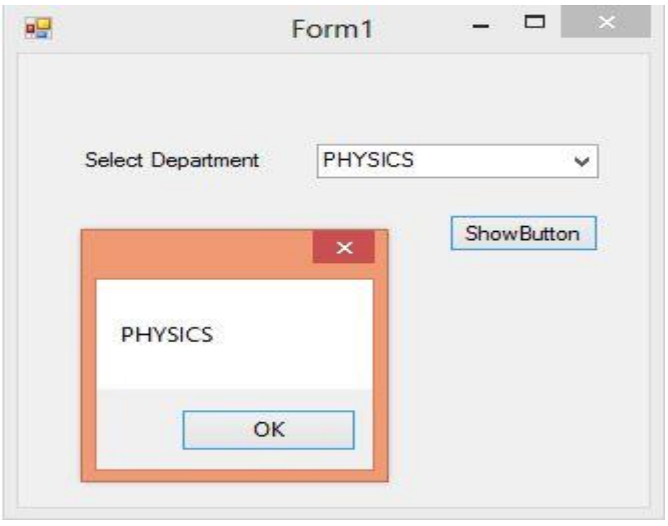
1. Create a desktop application and change it's properties(As like as Exercise-1)
2. Design the user interface like the following figure



3. Double click on the ShowButton
4. Write down the code snippet in the block

```
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;

namespace ComboBoxExample
{
public partial class Form1 : Form
 {
 public Form1()
   {
    InitializeComponent();

    comboload();  //Call your comboload() method
 }

private void comboload() // Create your own comboload() method
{
   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 =nameCB.FindStringExact("IIT");

}

private void button1_Click(object sender, EventArgs e)
{
    string dname= dnameCB.Text;
    MessageBox.Show(dname);
}
}
}
```

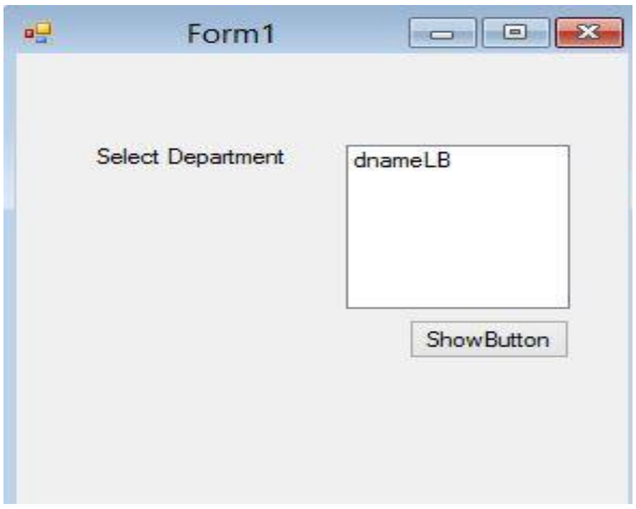
5. Build the solution and run it and You will get the following picture



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.

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



3. Double click on the ShowButton
4. Write down the code snippet in the block

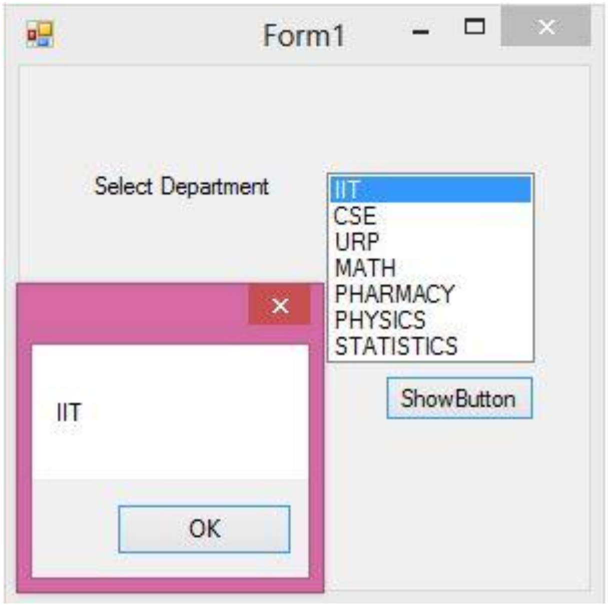
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace ListBoxControlExamples
{
public partial class Form1 : Form
 {
 public Form1()
   {
    InitializeComponent();

    listboxload();  //Call your listboxload method
}
```

```
private void listboxload() // Create your own load method
 {
     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;
}
```

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

 foreach (Object obj in dnameLB.SelectedItems)
 {
     MessageBox.Show(obj.ToString());
 }

}
```

```
}
}
```

5. Build the solution and run it and You will get the following picture

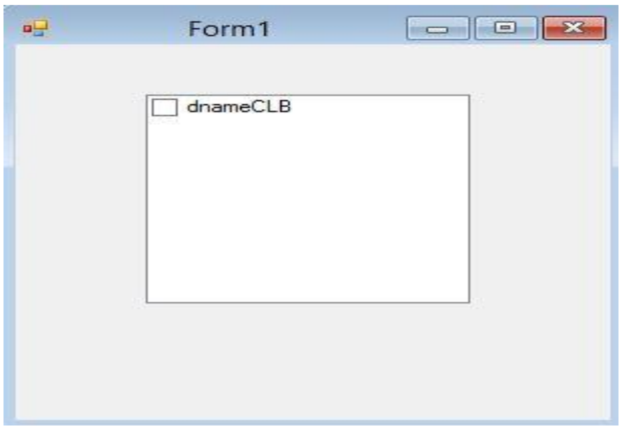


Exercise-4: Use of CheckedListBox Control

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.

1. Create a desktop application and change it’s properties(As like as Exercise-1)

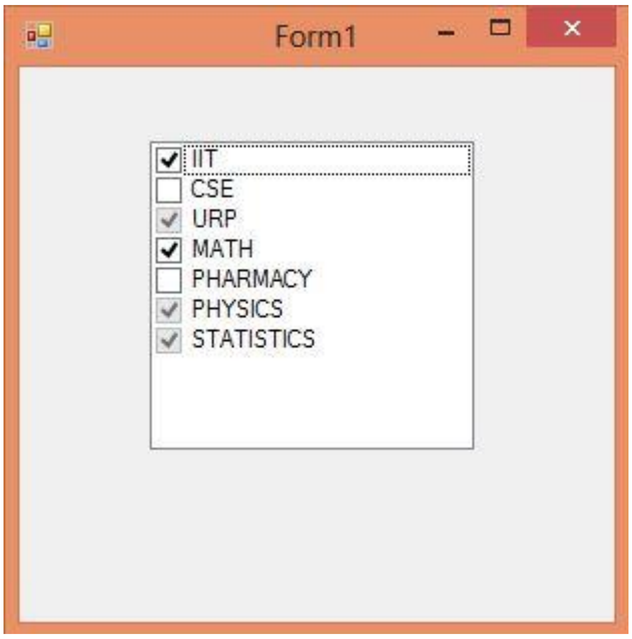
2. Design the user interface like the following figure



3. Create your own checklistbox() method
4. Call your `checklistboxload()` method in initialization state
5. Write down the code snippet

```
 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;

namespace CheckedListBoxexamples
 {
 public partial class Form1 : Form
  {
  public Form1()
     {
      InitializeComponent();

        checklistboxload(); //Call your  checklistboxload() method

     }

private void checklistboxload() // Create your own load method
 {
     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);


 }

   }
   }
```

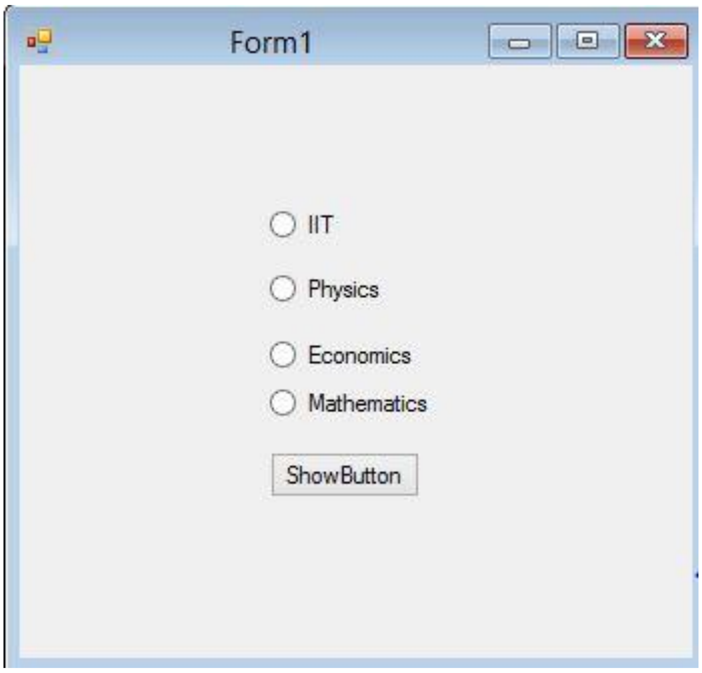
6. Build the solution and run it and You will get the following picture



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.

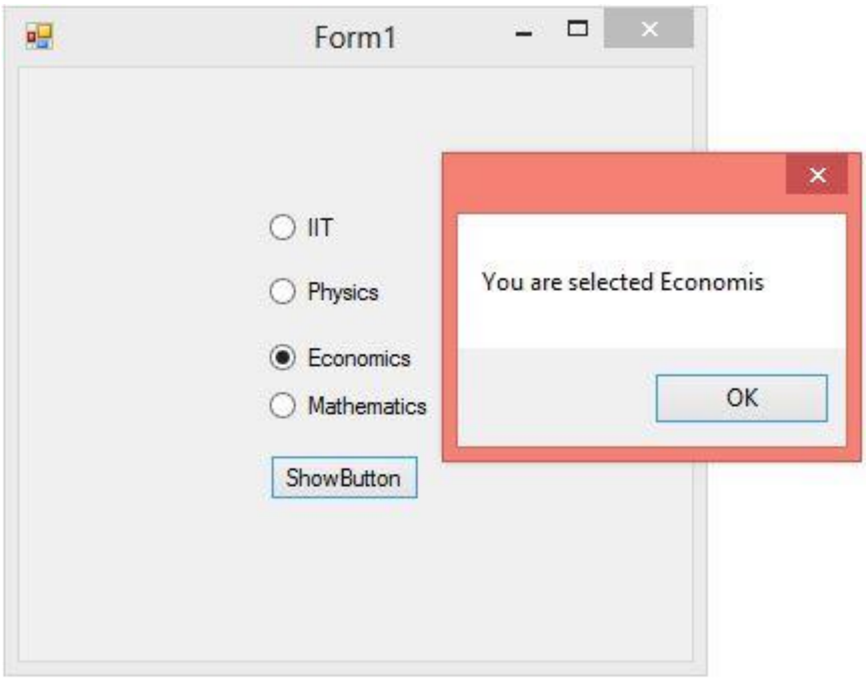
1. Create a desktop application and change it’s properties(As like as Exercise-1)
2. Design the user interface like the following figure



3. Double click on the ShowButton
4. Write down the code snippet in the block

```
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;

namespace RadioButtonExamples
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();

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

        }



    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;
         }

       }
}
}
```

5. Build the solution and run it and You will get the following picture



You are selected Economis

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.

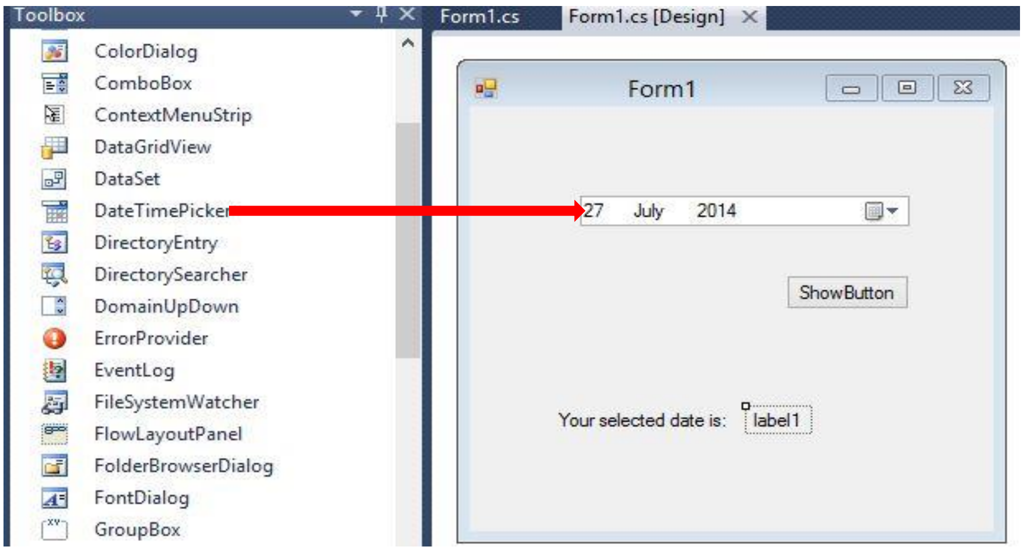
Practice-1: Create a desktop application by using CheckBox control.(As like as RadioButton Control)

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)

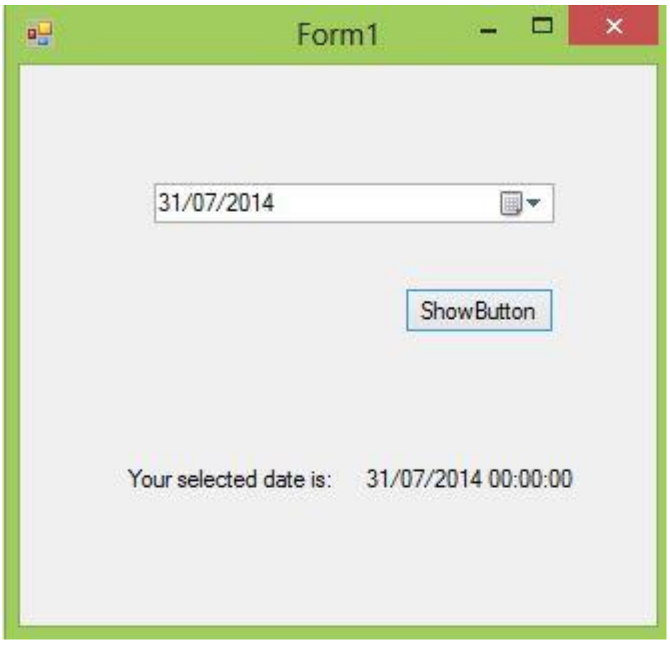
3. Design the user interface like the following figure



4. Double click on the ShowButton
5. Write down the code snippet in the block

```
 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;

namespace DatetimePickerExamples
{
public partial class Form1 : Form
 {
 public Form1()
  {
   InitializeComponent();

   // Initialize picker to yesterday.
   dateTP.Format = DateTimePickerFormat.Short;
   dateTP.Value = DateTime.Today;
}

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

 }

 }
 }
```

6. Build the solution and run it and here you will see output in the label and You will get the following picture



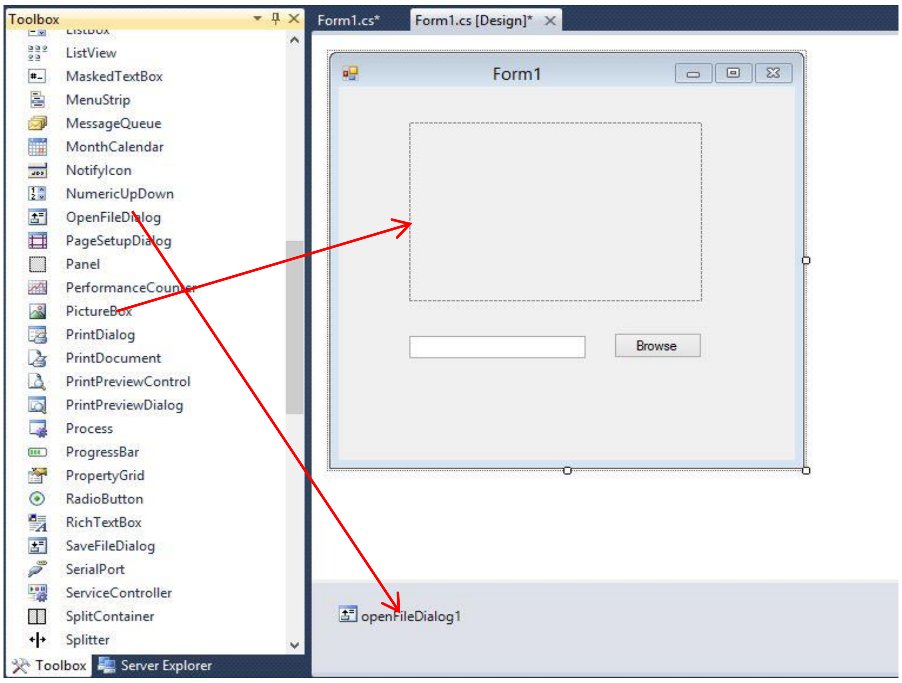
Your selected date is: 31/07/2014 00:00:00

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.

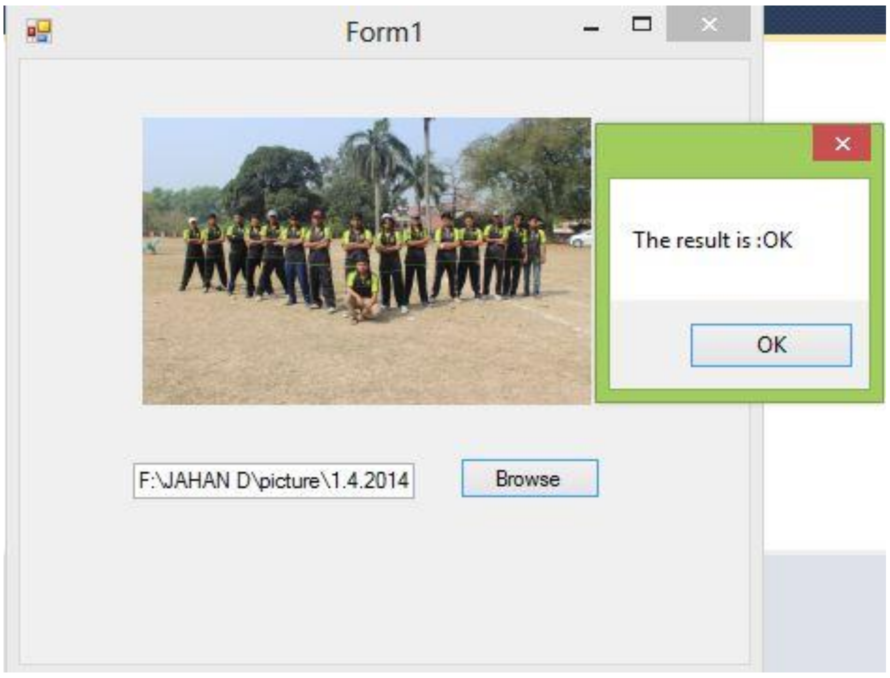
1. Create a desktop application (As like as Exercise-1)
2. Change the Name Property of PictureBox(must be select) into pictureBox and Change the Name and Text property of Button into browse_button ,Browse and Change the Name property of TextBox into filenameTB and openFileDialog1 is No need to change 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



4. Double click on the Browse Button
5. Write down the code snippet in the block

```
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.IO;
namespace OpenPictureExamples
  {
public partial class Form1 : Form
  {
public Form1()
 {
 InitializeComponent();
 }

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);
     }

       }
       }
```

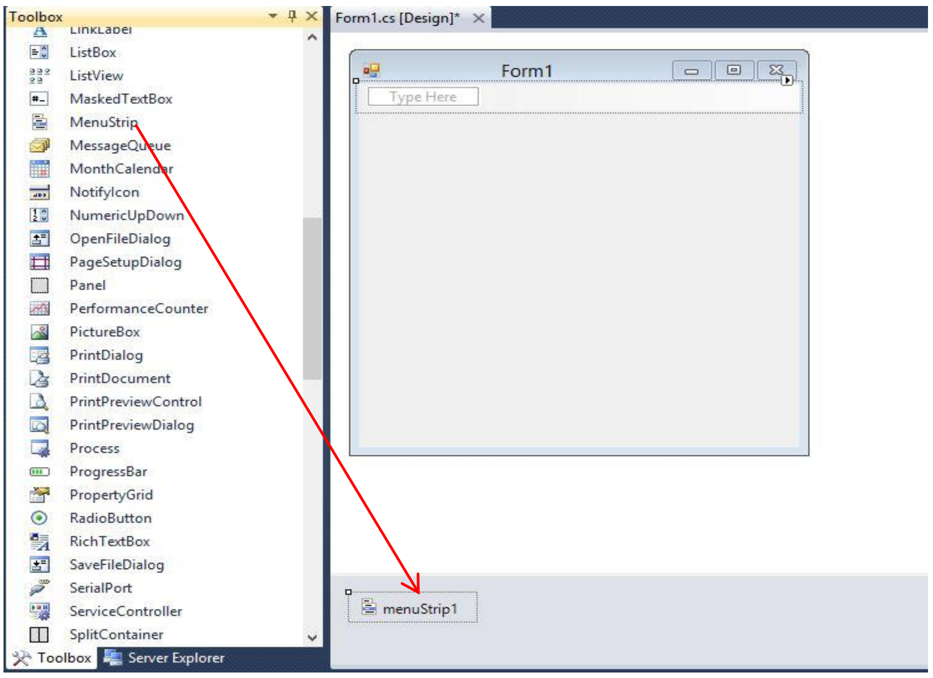
6. Build the solution and run it and You will get the following picture after browse



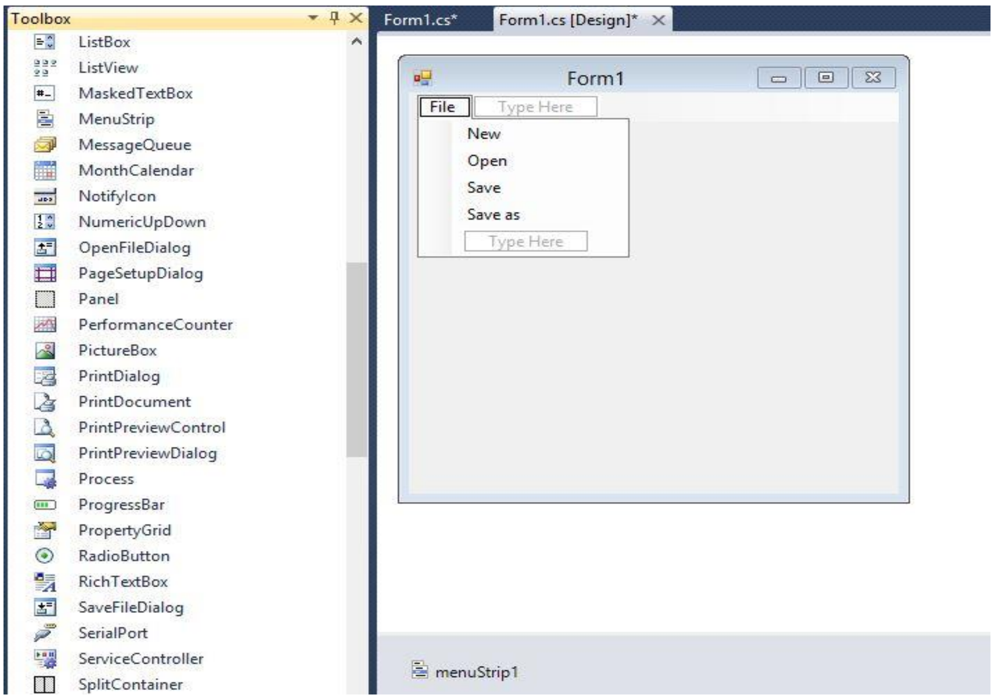
Exercise-9: Use of Menu Control

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.

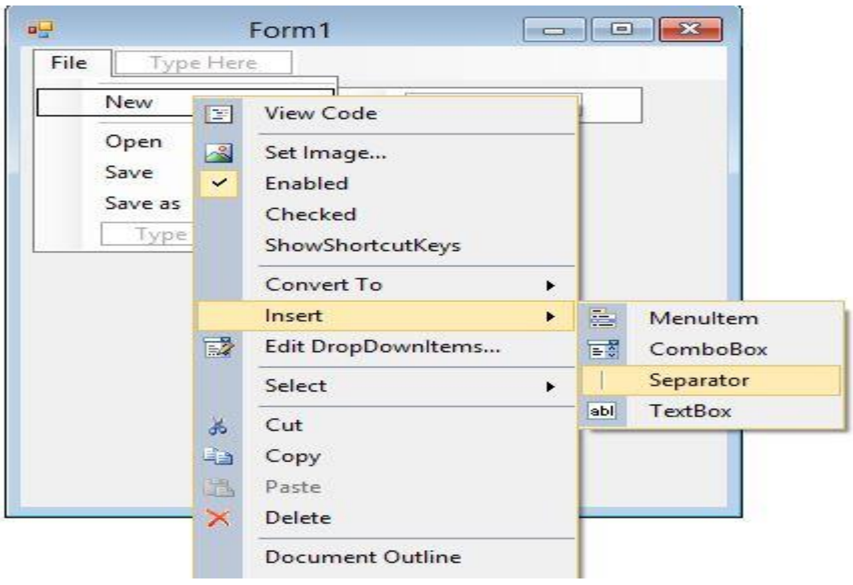
1. Create a desktop application
2. From the Toolbox, drag a MenuStrip control onto the form.



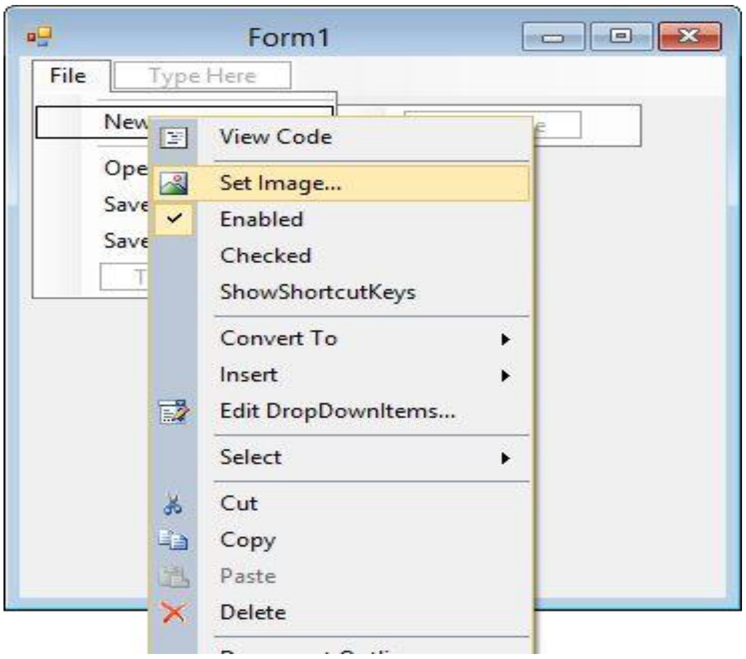
3. After drag the Menustrip on your form you can directly create the menu items by type a value into the "Type Here" box on the menubar part of your form. From the following picture you can understand how to create each menu items on mainmenu Object.



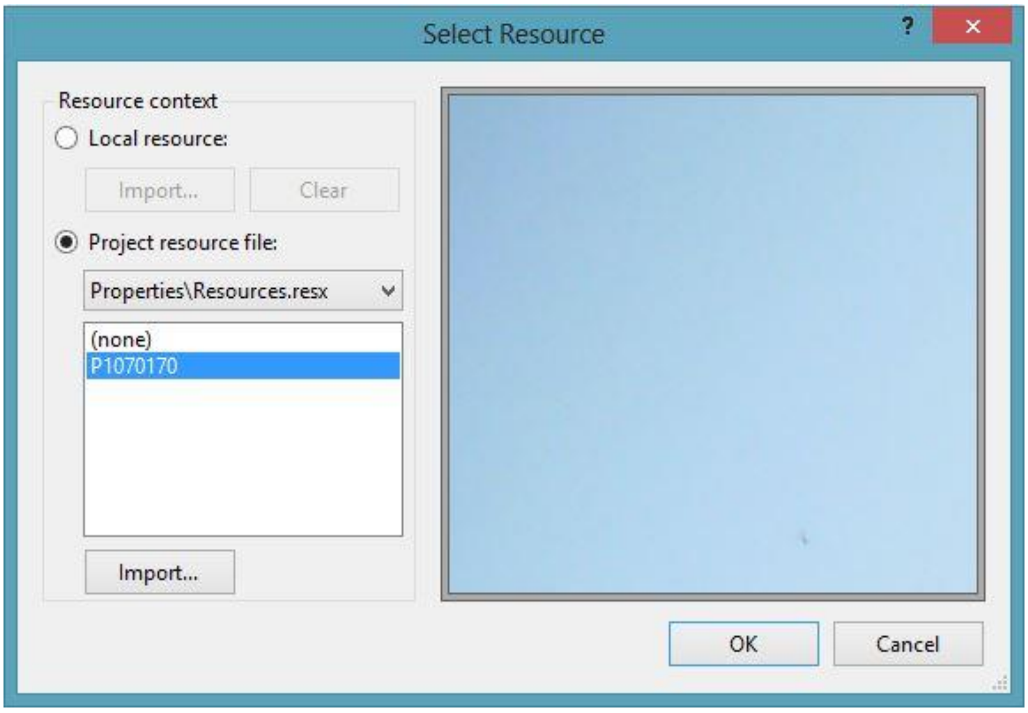
4. If you need a seperator bar , right click on your menu then go to insert->Seperator.



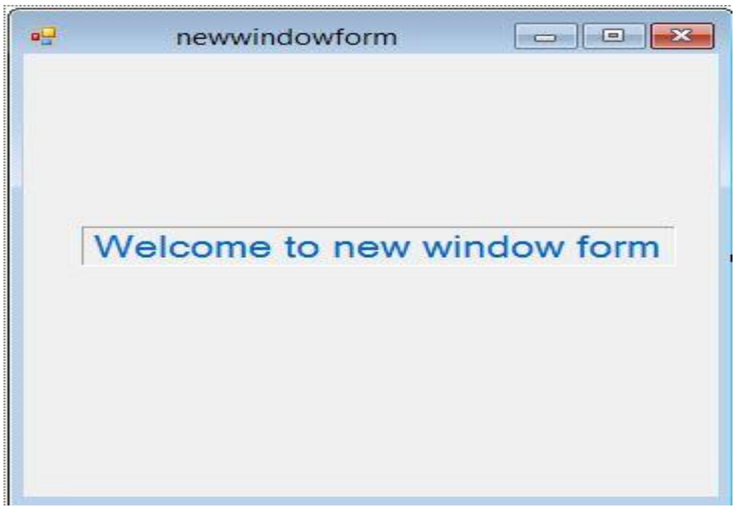
5. If you need a icon beside the menu item , right click on your menu then go to Set Image.



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.



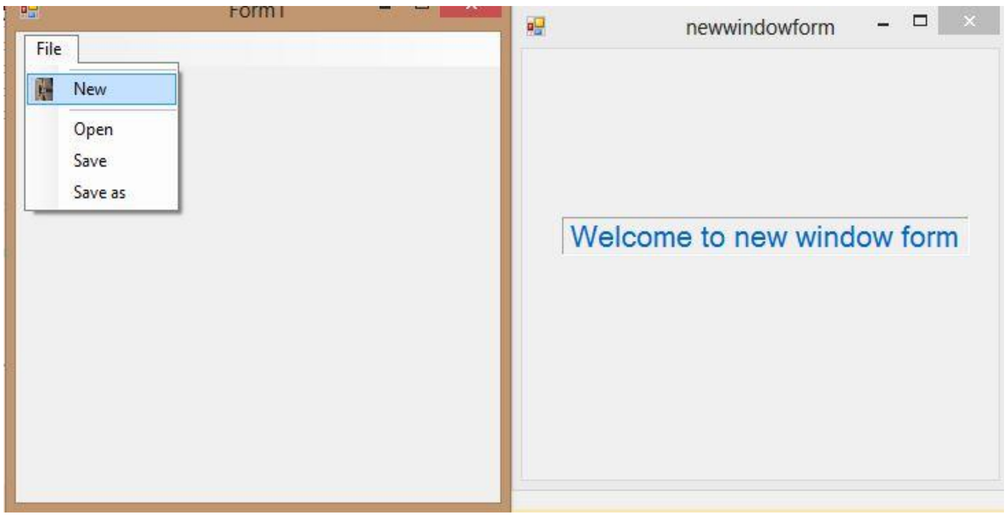
7. After creating the Menu on the form , you have to double click on each menu item and write the programs there depends on your requirements. Here we include a new window form.When we click New menu item then we will go to new window form.(See Task-2 ,How to add a new item or new window form) .



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

```
 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;

namespace menuExamples
 {
public partial class Form1 : Form
    {
    public Form1()
    {
        InitializeComponent();
    }

    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


Form1
File
New
Open
Save
Save as
newwindowform
Welcome to new window form