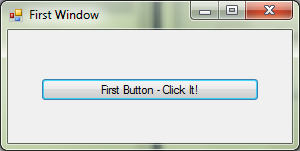
**Object-Oriented Application Development**

**Practical 9**

1. (a) Create a Windows-based application which has a graphical user interface as shown below (together with the names of the controls):



btnMyButton

MyForm

Set the properties for the controls using the Properties Window as follows:

|  |  |  |
| --- | --- | --- |
| **Control** | **Property** | **Value** |
| MyForm | Name  Text | MyForm  First Window |
| btnMyButton | Name  Text | btnMyButton  First Button – Click It! |

(b) Double-click on the *button* in the Form Designer View and add the code for the button-click event handler in the Code Editor View as follows:

|  |
| --- |
| private void btnMyButton\_Click(object sender, EventArgs e)  {  MessageBox.Show("You did it!!");  } |

Run the application and click on the button.

(c) In the Form Designer View, select the form and look at the Properties Window for the form. Click on the Events icon (lightning symbol). Look for the event FormClosing and double-click on it. Add the code for the form-closing event handler in the Code Editor View as follows:

|  |
| --- |
| private void MyForm\_FormClosing(object sender, FormClosingEventArgs e)  {  MessageBox.Show("The form will close now!!");  } |

Run the application and click on the X to close the window.

(d) In the Form Designer View, select the form and look at the Properties Window for the form. Look for the event Load and double-click on it. Add the code for the load event handler in the Code Editor View as follows:

|  |
| --- |
| private void MyForm\_Load(object sender, FormClosingEventArgs e)  {  MessageBox.Show("The form will open now!!");  } |

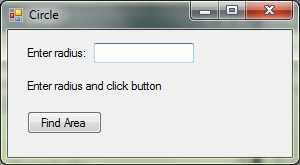
Run the application again.

(e) Look at the code and find the constructor for the form. Modify the constructor so that it looks like this.

|  |
| --- |
| public MyForm()  {  InitializeComponent();  btnMyButton.Enabled = false;  } |

Run the application again. *Note*: the button is now disabled.

1. Create a Windows-based application that computes the area of a circle. The application has a graphical user interface as shown below (together with the names of the controls):



tbxRadius

btnArea

lblMessage

lblRadius

CircleForm

Use the Properties Window to set the Name and Text properties of the controls as shown above.

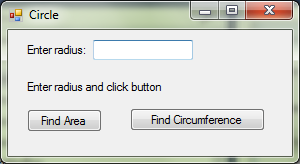
Add the following Circle class to the application:

|  |
| --- |
| public class Circle  {  private double radius;  public Circle(double theRadius)  { radius = theRadius; }  public double GetArea()  { return Math.PI \* Math.Pow(radius, 2); }  public double GetCircumference()  { return 2 \* Math.PI \* radius; }  } |

Double-click on the button on the form and add the following code for the button-click event handler:

|  |
| --- |
| private void btnArea\_Click(object sender, EventArgs e)  {  string input, result;  double radius, area;  input = tbxRadius.Text;  radius = Convert.ToDouble(input);  Circle c = new Circle(radius);  area = c.GetArea();  result = "The area of the circle is "  + string.Format("{0:F2}", area);  lblMessage.Text = result;  } |

1. Modify the Windows application in Question 2 to add a second button that allows the user to compute the circumference of a circle. Display the result in the label (lblMessage).



btnCircumference

tbxRadius

btnArea

lblMessage

lblRadius

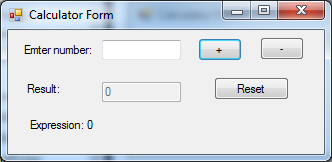
CircleForm

1. Create a Windows application that acts as a simple calculator which has a form as shown below. This is how the form looks like when it is first loaded.

btnSubtract

btnAdd

CalculatorForm



btnReset

lblExpression

tbxResult

tbxNumber

Consider the partial code for the application given below.

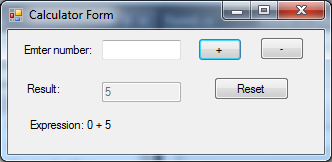
* There is an instance variable named ‘result’.
* When the application starts, the constructor is executed which initializes the instance variable ‘result’ and sets the text box ‘tbxResult’ to the string value of the instance variable ‘result’.
* The textbox ‘tbxResult’ is disabled by setting the ‘Enabled’ property to false.

|  |
| --- |
| public partial class CalculatorForm : Form  {  private int result; // result must be made an instance variable  public CalculatorForm() // executed when application starts  {  InitializeComponent();  tbxResult.Enabled = false; // disable the text box  result = 0;  tbxResult.Text = result.ToString();  }  private void btnAdd\_Click(object sender, EventArgs e)  {  // (a) write the code for this event-handler  int number = Convert.ToInt32(tbxNumber.Text);  result = result + number;  tbxResult.Text = result.ToString();  tbxNumber.Text = "";  lblExpression.Text = lblExpression.Text + "" + number;  }  private void btnSubtract\_Click(object sender, EventArgs e)  {  // (b) write the code for this event-handler  int number = Convert.ToInt32(tbxNumber.Text);  result = result - number;  tbxResult.Text = result.ToString();  tbxNumber.Text = "";  lblExpression.Text = lblExpression.Text + "" + number;  }  private void btnReset\_Click(object sender, EventArgs e)  {  // (c) write the code for this event-handler  result = 0;  tbxResult.Text = "0";  tbxNumber.Text = "";  lblExpression.Text = "Expression: 0";  }  } |

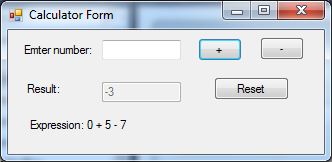
When the user enters a value for number and clicks on the ‘+’ or ‘-’ button, the application will:

* update ‘result’ and show the updated ‘result’ value in text box ‘tbxResult’
* clear the text box ‘tbxNumber’
* show an updated message in label ‘lblExpression’, with the action (+ or -) and the input number.

The forms below show some sample input and output.

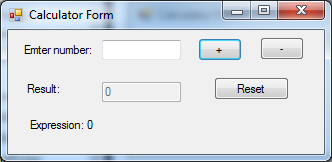


After the user enters ‘5’ and clicks ‘+’ button



After user enters ‘7’ and clicks ‘-’ button

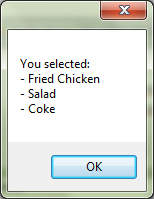
When the user clicks on the ‘Reset’ button, the application clears text box ‘tbxNumber’, sets the text in text box ‘tbxResult’ to 0, and sets the label ‘lblExpression’ to ‘Expression: 0’ as shown below.



After user clicks ‘Reset’ button

1. Create a Windows application that displays a form as shown below with the names of the controls given. After the user makes the selections and clicks on the button, the application displays the selected items using MessageBox.

(*Note*: Do NOT double-click on the check boxes or radio buttons in the Designer Window. Only double-click on the button to generate an event handler for button-click event.)

cbxChicken

cbxFries

cbxSalad

btnConfirm

rbnPepsi

rbnCoke

Windows form MessageBox