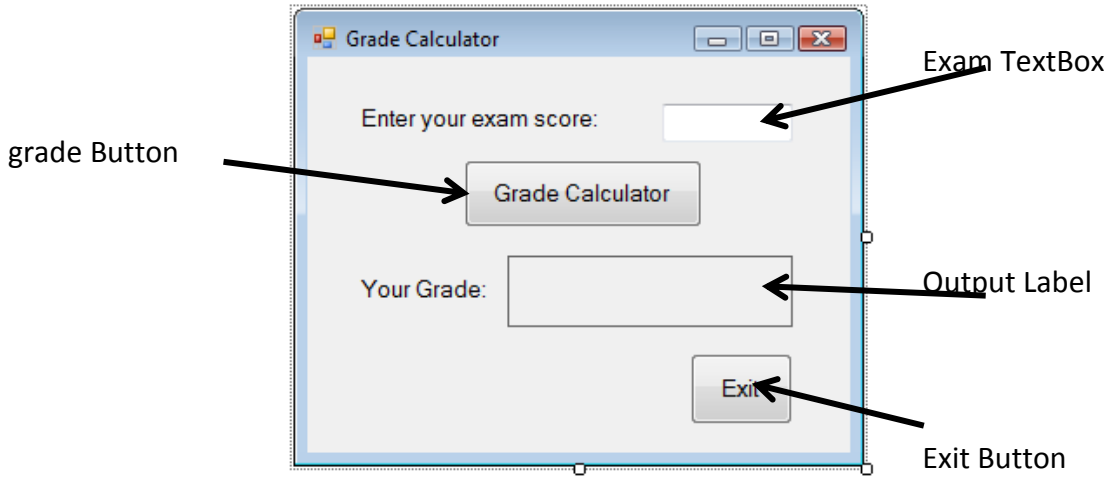
Making Decisions – (nested decision structures)

You will create an application that will get the user's exam score. It will use a **series of conditions** to determine if the user's grade is Honors (80 and above), Pass (60-80), or Fail (less than 60). It will display different messages depending on the user's exam mark.

Create a new Windows Form Application named **PassingGrade**

Create the GUI for the application as shown below:



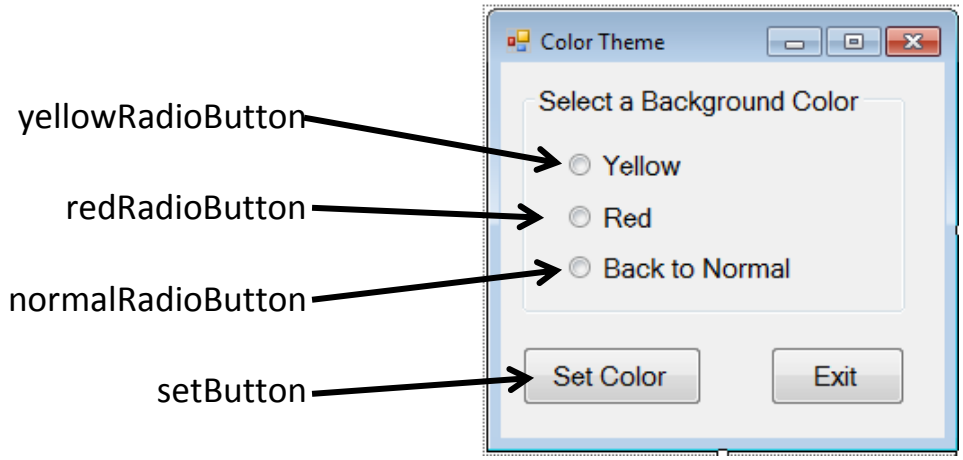
Create the Click Event handler for the displayButton

1. Declare the named constants
   - HONORS : 80
   - PASS : 60
2. Declare the local variables examScore
3. Get user input, convert (if necessary), store in variable
4. Determine if user grade is between 0 and 100, if false, show an error message if true:
   Determine if user grade is Honors. If true, display Honors message; if it is not true,
   determine if user grade is Pass If true, display Pass message. If false, display Fail message

Test the application.
Save All

Making Decisions –Radio Buttons & Check Boxes

Exercise 1: Create a new Windows Form Application named **BackgroundColor**.



Code needed to set background color of the form:

```
this.BackColor = Color.Yellow;
```

name of the color to be set

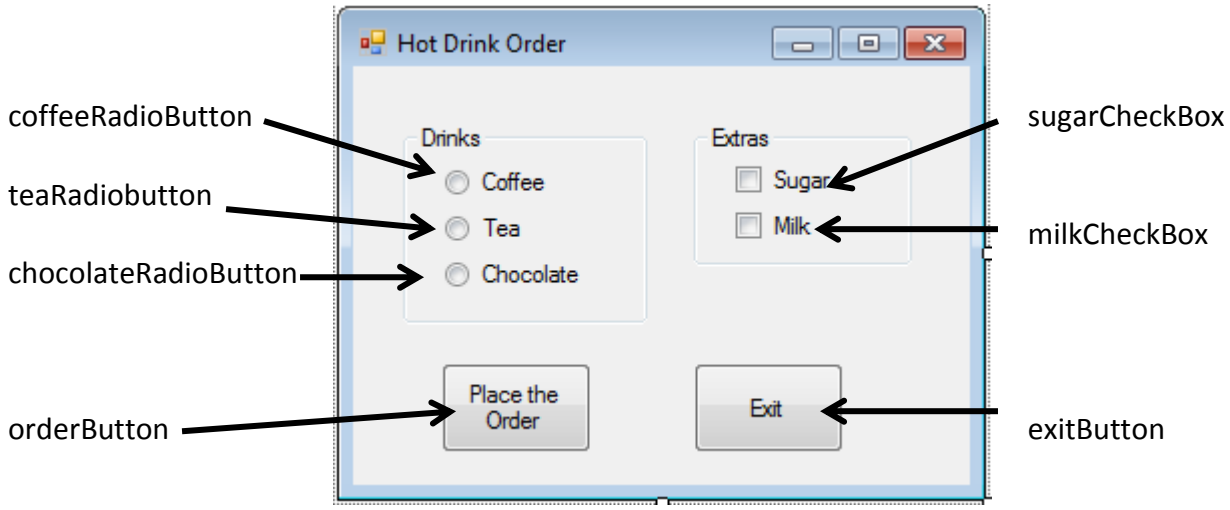
Create the Click Event handler for the setButton

5. Determine which radio button is checked & add item set the background color
   - if yellowRadioButton is checked, set color to yellow
   - else if redRadioButton is checked, set color to red
   - else normalRadioButton is checked, set color to the system color for the control

```
this.BackColor = SystemColors.Control;
```

```
private void setButton_Click(object sender, EventArgs e)
{
    //check if yellow button is selected; set color yellow
    if (yellowRadioButton.Checked)
    {
        this.BackColor = Color.Yellow;
    }
    //else if red button is selected; set color red
    else if (redRadioButton.Checked)
    {
        this.BackColor = Color.Red;
    }
    //else set color to system color
    else
    {
        this.BackColor = SystemColors.Control;
    }
}
```

Exercise 2: Create a new Windows Form Application named **RadioCheckbox**



Create the Click Event handler for the orderButton

6. Declare the local variables and set starting values
   - orderMessage = "You ordered: \n"
   - extraMessage = " with \n"
7. Determine which radio button is checked & add item to the orderMessage (**+=**)
   - if coffeeRadioButton is checked, add "Coffee"
   - else if teaRadioButton is checked, add "Tea"
   - else if chocolateRadioButton is checked, add "Chocolate"
   - else add "You have not made a selection"
8. Determine if each check box is checked & add item to extraMessage (**+=**)
   - if sugarCheckBox is checked, add " sugar \n"
   - if milkCheckBox is checked, add " milk \n"
9. Determine if NO boxes are checked
   - if sugarCheckBox NOT checked AND milkCheckBox NOT checked, add "no extras.
10. Display the full message combining order message and extraMessage

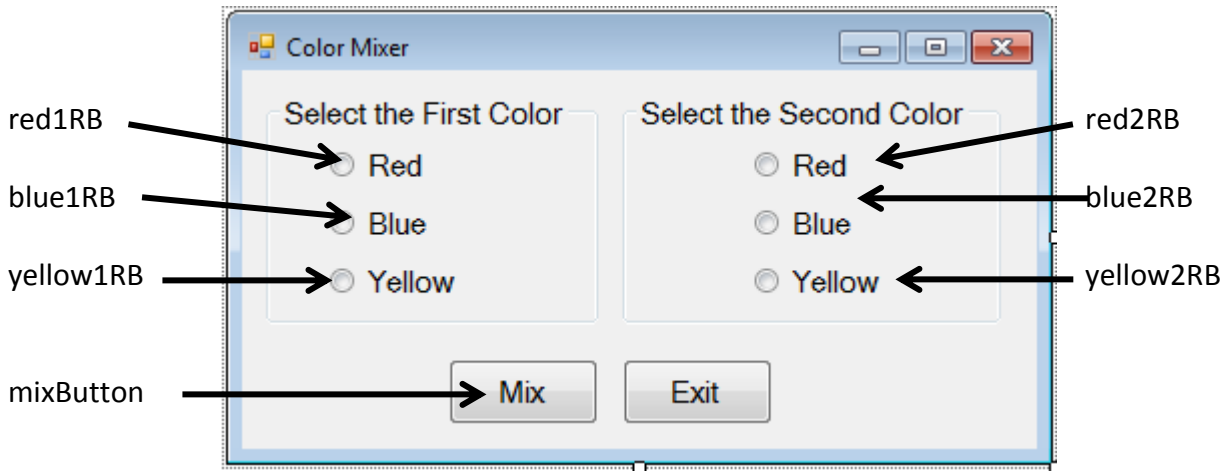
```
private void orderButton_Click(object sender, EventArgs e)
{
    //declare variable to accumulate & store the order
    string orderMessage = "You ordered:  \n";
    string extraMessage = " with \n";

    //use if/else if to identify selection and store in orderMessage
    if (coffeeRadioButton.Checked)
    {
        orderMessage += "Coffee";
    }
    else if (teaRadioButton.Checked)
    {
        orderMessage += "Tea";
    }
    else if (chocolateRadioButton.Checked)
    {
        orderMessage += "Chocolate";
    }
    else
    {
        orderMessage += "You have not made a selection";
    }


    //use series of if statements to identify any extras chosen
    //and store in extraMessage
    if (sugarCheckBox.Checked)
    {
        extraMessage += "  sugar\n";
    }
    if (milkCheckBox.Checked)
    {
        extraMessage += "  milk\n";
    }

    //use if with logical operators to see if no checkboxes are checked
    if (!(sugarCheckBox.Checked) && !(milkCheckBox.Checked))
    {
        extraMessage += "  no extras";
    }

    //display order using orderMessage and extraMessage
    MessageBox.Show(orderMessage + extraMessage);
}
```

Exercise 3: Create a new Windows Form Application named **ColorMixer**.

The colors red, blue, and yellow are known as *primary colors*. When you mix two primary colors you get a secondary color:

- mix red and blue to get purple
- mix red and yellow to get orange
- mix blue and yellow to get green

Your application should let the user select two primary colors from two different sets of *Radio* buttons. When the user clicks the *Mix* button, the form's background color should change to the color you get when you mix the two primary colors. If the user selects that same color from both sets of *Radio* buttons, set the form's background to that color.



Create the Click Event handler for the mixButton

1. Check if same colors are chosen and set the background to that color
   - if red1RB is checked AND red2RB is checked, set color to red
   - else if blue1RB is checked AND blue2RB is checked, set color to blue
   - else if yellow1RB is checked AND yellow2RB is checked, set color to yellow
2. Else Check if red and blue are chose, then set to purple
   - else if (red1RB AND blue2RB) OR (red2RB AND blue1RB), set color to purple
3. Else Check if red and yellow are chose, then set to orange
   - else if (red1RB AND yellow2RB) OR (red2RB AND yellow1RB), set color to orange
4. Else Check if blue and yellow are chose, then set to green
   - else if (blue1RB AND yellow2RB) OR (blue2RB AND yellow1RB), set color to green
5. Else set the color to the system color

```
private void mixButton_Click(object sender, EventArgs e)
{
    //check if same colors are chosen & set the color
    if (red1RB.Checked && red2RB.Checked)
    {
        this.BackColor = Color.Red;
    }
    else if (blue1RB.Checked && blue2RB.Checked)
    {
        this.BackColor = Color.Blue;
    }
    else if (yellow1RB.Checked && yellow2RB.Checked)
    {
        this.BackColor = Color.Yellow;
    }
    //check for red and blue = purple
    else if ((red1RB.Checked && blue2RB.Checked) || (red2RB.Checked && blue1RB.Checked))
    {
        this.BackColor = Color.Purple;
    }
    //check for red and yellow = orange
    else if ((red1RB.Checked && yellow2RB.Checked) || (red2RB.Checked && yellow1RB.Checked))
    {
        this.BackColor = Color.Orange;
    }
    //check for yellow and blue = green
    else if ((yellow1RB.Checked && blue2RB.Checked) || (yellow2RB.Checked && blue1RB.Checked))
    {
        this.BackColor = Color.Green;
    }
    //else set to system color
    else
    {
         this.BackColor = SystemColors.Control;
    }
}
```