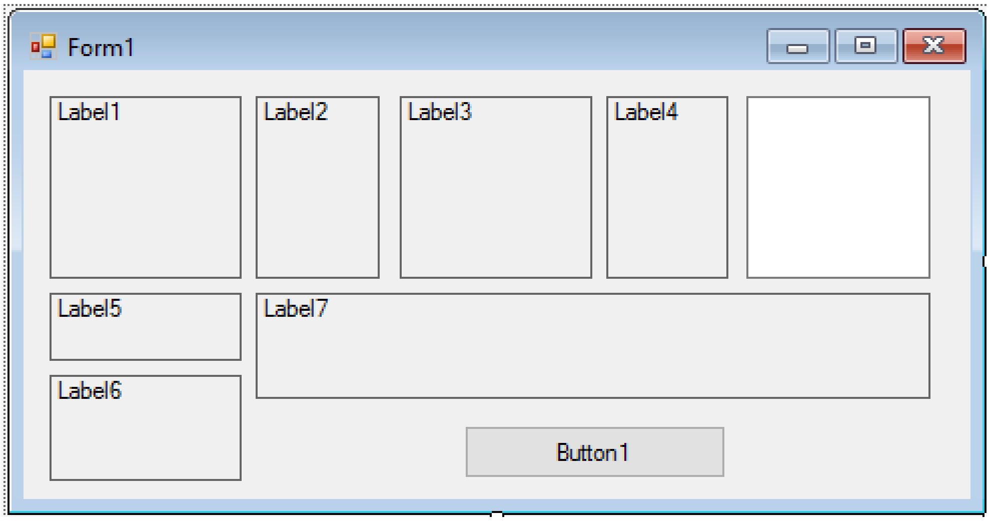
Project 2 – Times Tables

Project Design

In this project, you can give a child practice with the times tables using the numbers from 0 to 9. The computer generates a random problem. The child answers and the computer evaluates the performance. The project you are about to build is saved as **Times** in the project folder (**\BeginVCS\BVCS Projects**).

Place Controls on Form

Start a new project in Visual C#. Place seven labels (with **AutoSize** set to **False**, to allow resizing), a text box and a button on the form. When done, your form should look something like this (I've temporarily set the border style of each label control to **FixedSingle** to show placement; you might also like to do this, but remember to change border style back to None):



Set Control Properties

Set the control properties using the properties window:

Form1 Form:

Property Name	Property Value
Text	Times Tables
FormBorderStyle	FixedSingle
StartPosition	CenterScreen

label1 Label:

Property Name	Property Value
Name	lblNum1
Text	[Blank]
TextAlign	MiddleCenter
Font	Arial
Font Size	48
AutoSize	False

label2 Label:

Property Name	Property Value
Text	x
TextAlign	MiddleCenter
Font	Arial
Font Size	48
AutoSize	False

label3 Label:

Property Name	Property Value
Name	lblNum2
Text	[Blank]
TextAlign	MiddleCenter
Font	Arial
Font Size	48
AutoSize	False

label4 Label:

Property Name	**Property Value**
Text	=
TextAlign	MiddleCenter
Font	Arial
Font Size	48
AutoSize	False

label5 Label:

Property Name	**Property Value**
Text	Score:
TextAlign	MiddleCenter
Font Size	18
AutoSize	False

label6 Label:

Property Name	**Property Value**
Name	lblScore
Text	0%
TextAlign	MiddleCenter
BackColor	Light Yellow
BorderStyle	Fixed3D
Font Size	20
AutoSize	False

label7 Label:

Property Name	**Property Value**
Name	lblMessage
Text	[Blank]
TextAlign	MiddleCenter
BackColor	Light Yellow
BorderStyle	Fixed3D
Font Size	24
AutoSize	False

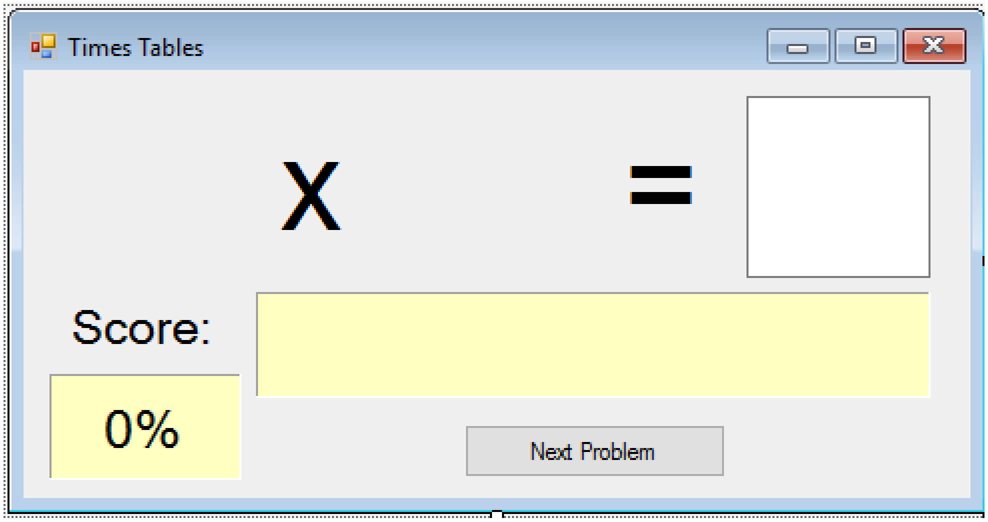
textBox1 Text Box:

Property Name	**Property Value**
Name	txtAnswer
Text	[Blank]
TextAlign	Center
Font	Arial
Font Size	48
MaxLength	2

button1 Button:

Property Name	**Property Value**
Name	btnNext
Text	Next Problem

When done setting properties, my form looks like this:



Write Event Methods

When the user clicks **Next Problem**, the computer generates and displays a multiplication problem. The user types an answer and presses **<Enter>.** If correct, you are told so. If incorrect, the correct answer is given. In either case, the score is updated. Continue answering as long as you would like.

Add this code to the **general declarations** area:

```
int product;
int numProb;
int numRight;
Random myRandom = new Random();
```

The **Form1_Load** event method:

```
private void Form1_Load(object sender, EventArgs e)
{
    // Initialize variables
    numProb = 0;
    numRight = 0;
    // display the first problem
    btnNext.PerformClick();
}
```

The **btnNext_Click** event method:

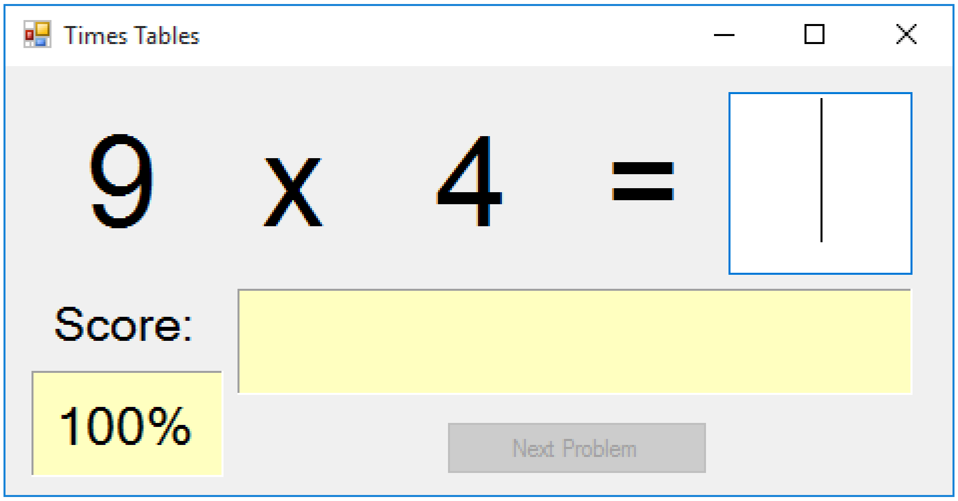
```
private void btnNext_Click(object sender, EventArgs e)
{
    // Generate next multiplication problem
    int number1, number2;
    txtAnswer.Text = "";
    lblMessage.Text = "";
    numProb++;
    // Generate random numbers for factors
    number1 = myRandom.Next(10);
    number2 = myRandom.Next(10);
    lblNum1.Text = Convert.ToString(number1);
    lblNum2.Text = Convert.ToString(number2);
    // Find product
    product = number1 * number2;
    btnNext.Enabled = false;
    txtAnswer.Focus();
}
```

The **txtAnswer_KeyPress** event method:

```
private void txtAnswer_KeyPress(object sender,
KeyPressEventArgs e)
{
    int ans;
    // Check for number only input and for return key
    if ((e.KeyChar >= '0' && e.KeyChar <= '9') || (int)
e.KeyChar == 8)
    {
        e.Handled = false;
    }
    else if ((int) e.KeyChar == 13)
    {
        // Check answer and update score
        ans = Convert.ToInt32(txtAnswer.Text);
        if (ans == product)
        {
            numRight++;
            lblMessage.Text = "That's correct!";
        }
        else
        {
            lblMessage.Text = "Answer is " +
Convert.ToString(product);
        }
        lblScore.Text = String.Format("{0:f0}", 100 *
((double) numRight / numProb)) + "%";
        btnNext.Enabled = true;
        btnNext.Focus();
    }
    else
    {
        e.Handled = true;
    }
}
```

Run the Project

Save your work. Run the project. A multiplication problem will be displayed. Type an answer and press <Enter>. If correct, that's great. If not, you will be shown the correct answer. Click **Next Problem** for another problem. Try for a high score. Here's a run I made:



Other Things to Try

Some suggested changes to make this a more useful program are: (1) make the range of factors an option (small numbers for little kids, large numbers for older kids), (2) allow practice with a specific factor only, (3) give the user more chances at the correct answer with a decreasing score for each try, (4) set up a timer so the faster the user answers, the higher the score and (5) expand the program to include other operations such as addition, subtraction and division.