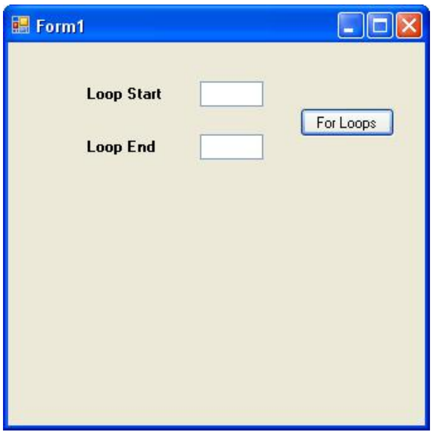
C# Loops and IF recap- based on tutorials from homeandlearn

Start a new project that is a windows form application. Add two text boxes to your form. Add a couple of labels and a button as well. For the first label, type Loop Start. For the second label, type Loop End. Your form will then look something like this:



What we'll do is to get the start value and end value from the text boxes. We'll then use these in our for loop.

So double click your button to get at the code (or press F7 on your keyboard). Set up two variables to hold the numbers from the text boxes:

int loopStart;

int loopEnd;

Now store the numbers from the text boxes into the two new variables:

loopStart = int.Parse(textBox1.Text);

loopEnd = int.Parse(???.Text);

Task 1: replace ??? in your code with the name of the textbox you're getting the value of loopEnd from (think.

Now that we have the numbers from the text boxes, we can use them in the for loop.

Create a for loop that looks like this (between the button click brackets):

```
int answer =0;

        for (int i = loopStart; i < loopEnd; i++)

        {

                answer = answer + i;

        }

        MessageBox.show(answer);
```

So instead of storing a value of 1 in the variable called i, we've stored whatever is in the variable called **loopStart**. Whatever you type in the first text box is now used as the starting value of the loop.

We're using the value stored inside of **loopEnd**. We're telling C# to keep looping if the value inside of the i variable is less than **loopEnd**. (Remember, because our Update Expression is

i++, C# will keep adding 1 to the value of i each time round the loop. When i is no longer less than the value of **loopEnd**, C# will stop looping.)

Run your program and type 1 in the first text box and 10 in the second text box. Click your button. You should find that the message box displays an answer of 45.

Can you see a problem here? If you wanted to add up the numbers from 1 to 10, then the answer is wrong! It should be 55, and not 45.

Task 2:- Can you see why 45 is displayed in the message box, and not 55? Fix this problem if you can.

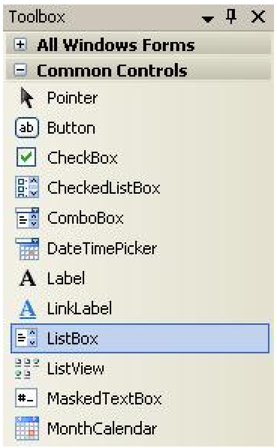
(There is, another problem. If you don't type anything at all in the text boxes, your program will crash! It does this because, you can't expect C# to convert something that's not there.

Task 3 :- Can you think of a way to solve this using if statements?

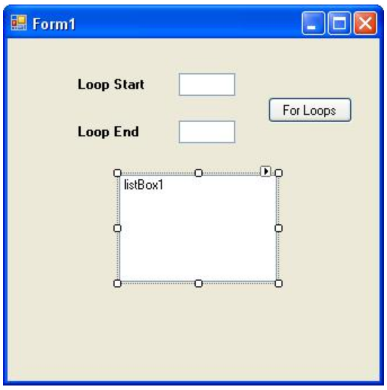
We can now write a little times table programme. We'll use the text boxes to ask users to input a start number and end number. We'll use these to display the 10 times table. So if the user types a 1 into the first text box and a 5 into the second text box, we'll display this:

1 times 10 = 10
2 times 10 = 20
3 times 10 = 30
4 times 10 = 40
5 times 10 = 50

Instead of using a message box to display the results, we'll use a List Box. A list box, you will not be surprised to hear, is used to display lists of items. But it's easier to show you what they do rather than explain. So use the Toolbox on the left of Visual C# to add a list box to your form:



Resize your list box, and your form will now look something like this:



Double click your button to get at your code. Now add this line to your loop (the line to add is in italics below):

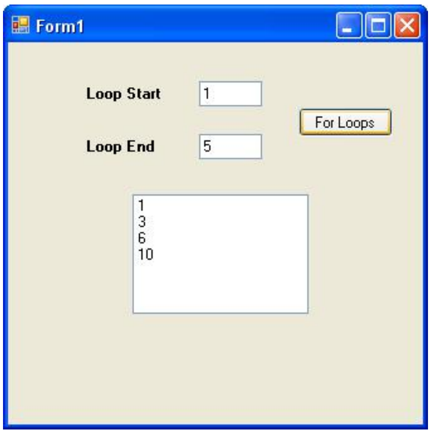
```
for (int i = loopStart; i <= loopEnd; i++)
{
            answer = answer + i;
            listBox1.Items.Add( answer.ToString() );
}
```

So you start by typing the Name of your list box (**listBox1**, for us). After a dot, you should see the IntelliSense list appear. Select **Items** from the list. **Items** is another property of list boxes. It refers to the items in your list. After the word Items, you type another dot. From the IntelliSense list, select the

C# Loops and IF recap- based on tutorials from homeandlearn

Add method. As its name suggests, the Add method adds items to your list box. Between the round brackets, you type what you want to add to the list of items. In our case, this was just the answer, converted to a string.

You can delete the message box line, if you like, because you don't need it. But run your program and enter 1 in the first text box and 5 in the second text box. Click your button and your form should look like this:



Loop Start 1

Loop End 5

```
1
3
6
10
```

The program is supposed to add up the number 1 to 5, or whatever numbers were typed in the text boxes. The list box is displaying one answer for every time round the loop. However, it's only displaying 4 items. If you solved the problem as to why 45 was displayed in the message box, and not 55 then you'll already know why there are only four items in the list box. If you didn't, examine the first line of the for loop:

for (int i = loopStart; i < loopEnd; i++)

The problem is the second part of the loop code: **i < loopEnd**

We're telling C# to go round and round while the value in **i** is less than the value in **loopEnd**. C# will stop looping when the values are equal. The value in loopEnd is 5 in our little program. So we're saying to C#, "Keep looping while the value in i is less than 5. Stop looping if it's 5 or more."

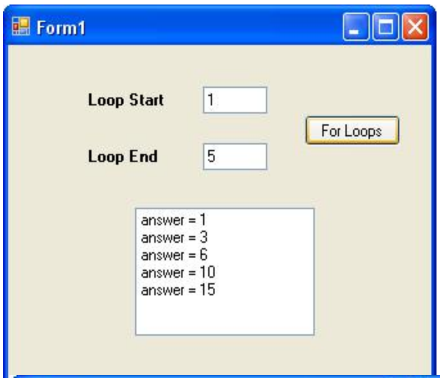
Task 4

Cleary, we've used the wrong Conditional Operator. Instead of using the less than operator, we need … well, which one do we need? Replace the < symbol with the correct one.

To add some more information in your list box, change your line of code to this:

listBox1.Items.Add("answer = " + answer.ToString( ));

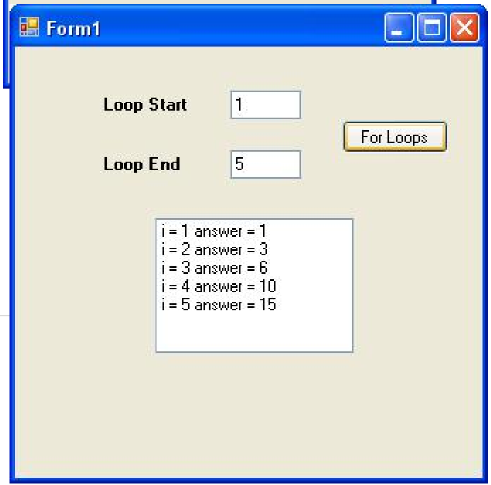
The thing to add is the text in red above. We've typed some direct text **"answer="** and followed this with the concatenation symbol (**+**). C# will then join the two together, and display the result in your list box. Run your programme again, and the list box will be this:



Loop Start 1

Loop End 5

```
answer = 1
answer = 3
answer = 6
answer = 10
answer = 15
```



Loop Start 1

Loop End 5

```
i = 1 answer = 1
i = 2 answer = 3
i = 3 answer = 6
i = 4 answer = 10
i = 5 answer = 15
```

To make it even clearer, add some more text to your list box. Try this: **listBox1.Items.Add(** ***"i = " + i +*** **" answer = " +**

answer.ToString( )); Again, the text to add is in italics.

(This code takes the string i, then adds to the end of the string the value of the variable i, then adds to the end the text “answer =” then the value of the variable answer turned into a string.

Run your program, and your list box will look like this:

<u>The Times Table Programme</u>

We now have all the ingredients to write a Times Table program.

What we're going to do, is to use a for loop to calculate and display the 10 times table. We need another variable, to hold the 10. So add this variable to your code: **int multiplyBy = 10;**

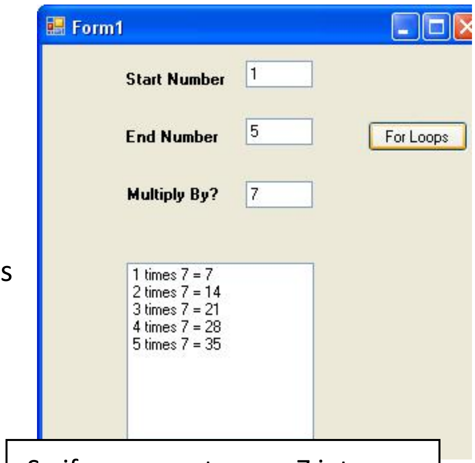
The only other thing we need to do is to change the code between the curly brackets of the for loop. At the moment, we have this:

```
answer = answer + i;
listBox1.Items.Add( "i = " + i + " answer = " + answer.ToString() );
```

Delete these two lines and replace them with these two:

```
answer = multiplyBy * i;
listBox1.Items.Add(i + " times " + multiplyBy + " = " + answer.ToString());
```

The first line of the code, is a simple multiplication. We multiply whatever is inside of the variable called **multiplyBy** (which is 10) by whatever is inside of the variable called **i**. C# is adding 1 to the value of **i** each time round the loop, so the answer gets updated and then displayed in the list box.



Start Number 1
End Number 5
Multiply By? 7

```
1 times 7 = 7
2 times 7 = 14
3 times 7 = 21
4 times 7 = 28
5 times 7 = 35
```

So if your user types a 7 into your new text box, clicking the button would display the 7 times table in the list box. Here's what your program should look like when you're finished

Run your program. Enter 1 in the first text box and 10 in the second text box. Click to see the ten times table:

At the moment, we're multiplying by 10 and, therefore, displaying the 10 times table.
Task 5: Add another text box to your form. Add a label that asks users which times table they want. Use this value in your code to display the times table for them.

Task 6: use **listBox1.Items.Clear()**; to clear the value of the clear whenever the button is pressed.