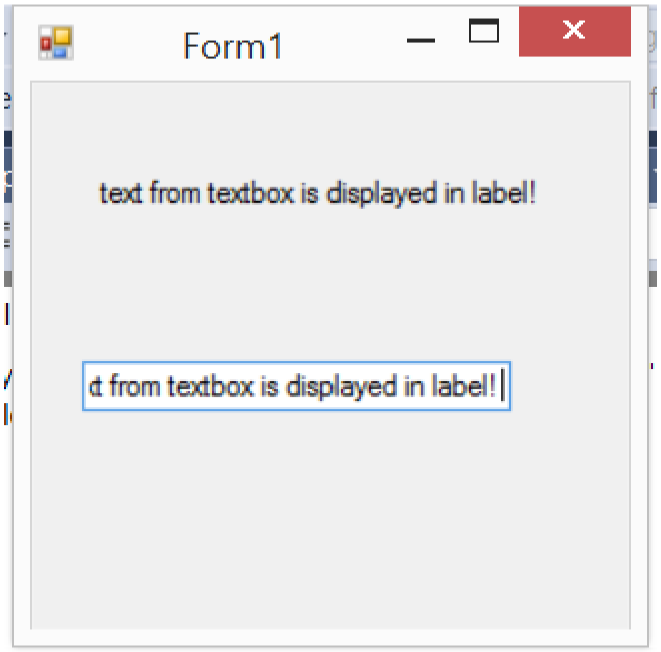
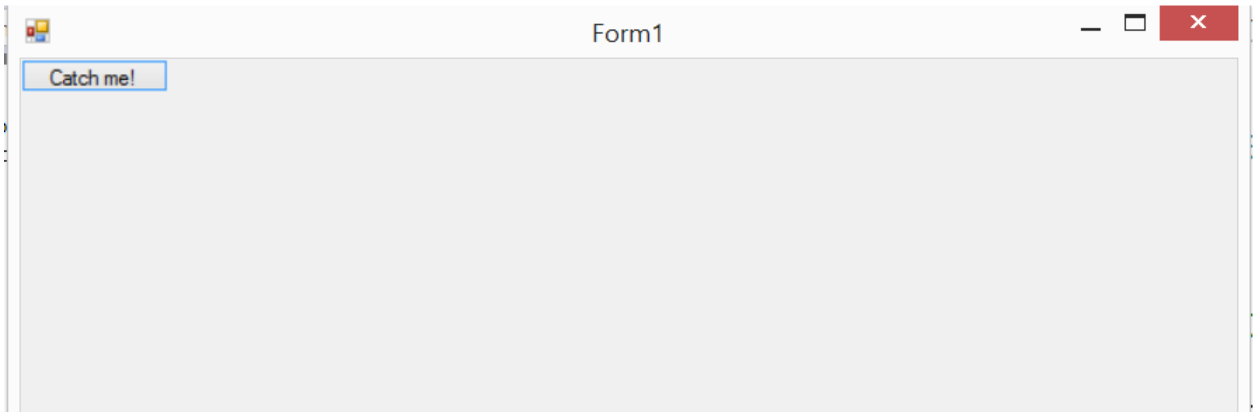
Practice exercises: do not submit them!

1. Add a text box and a label on the form. Change the label text every time something is typed in the text box.
   - Tip: text box has an event called „`TextChanged`“ which is fired (executed) EVERY TIME the text in the text box changes. This event can be generated by double clicking on a text box.


Form1
text from textbox is displayed in label!
t from textbox is displayed in label!

2. Add one button to the form and add text „Catch me!“. When form is loaded, change the location of the button randomly after every 0.5 seconds (so the user has to chase it to click it). When user clicks on the button then stop moving it around and change the text on the button.

   Tip: Positions of all controls can be changed by using *control.Top* and *control.Left* properties.
   Form height and Form width can be received by using Height and Width properties.
   Example of a button with properties *button1.Top=0* and *button1.Left=0*

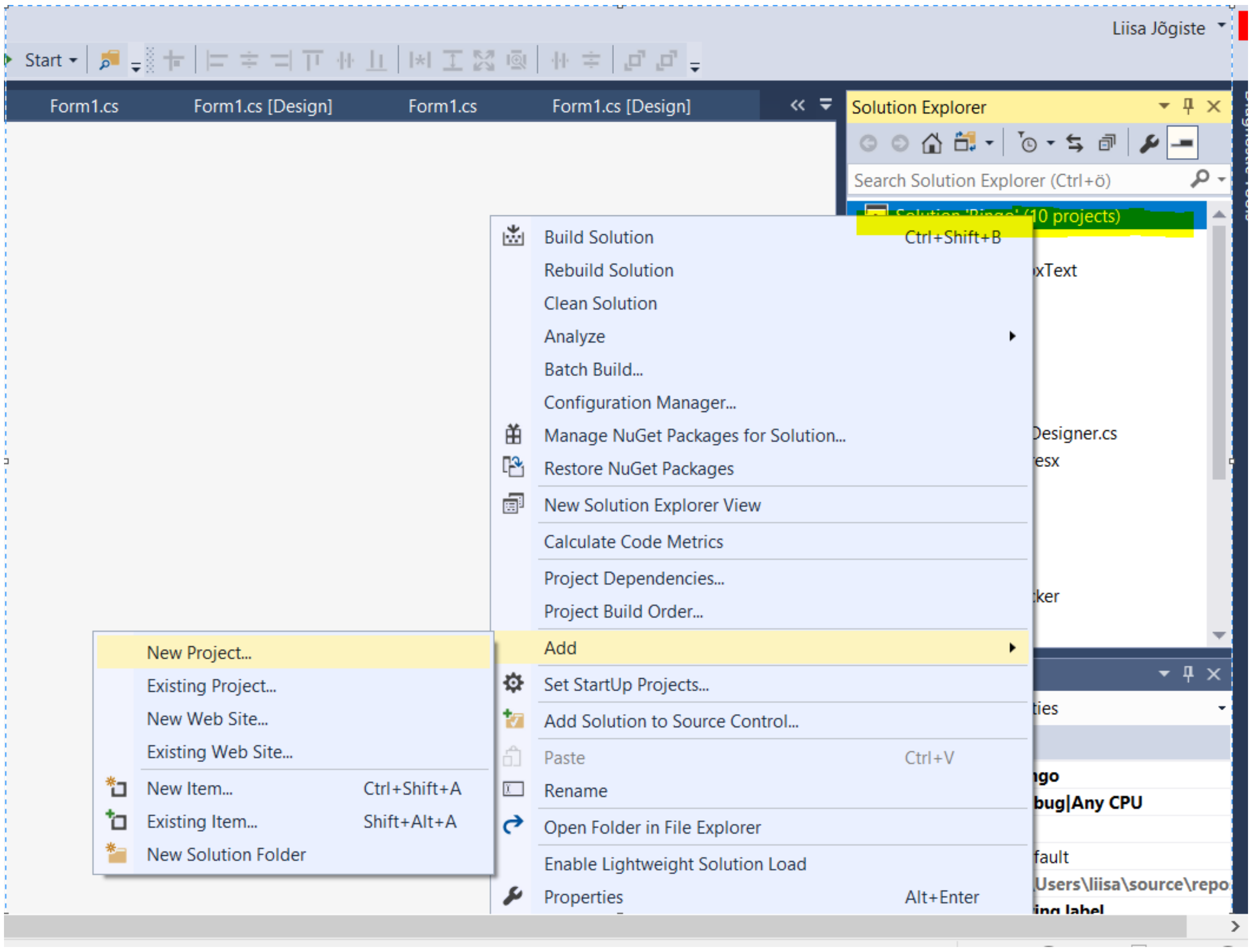


These exercises have solutions in ained.ttu.ee page. Try to do them and then check the solution.

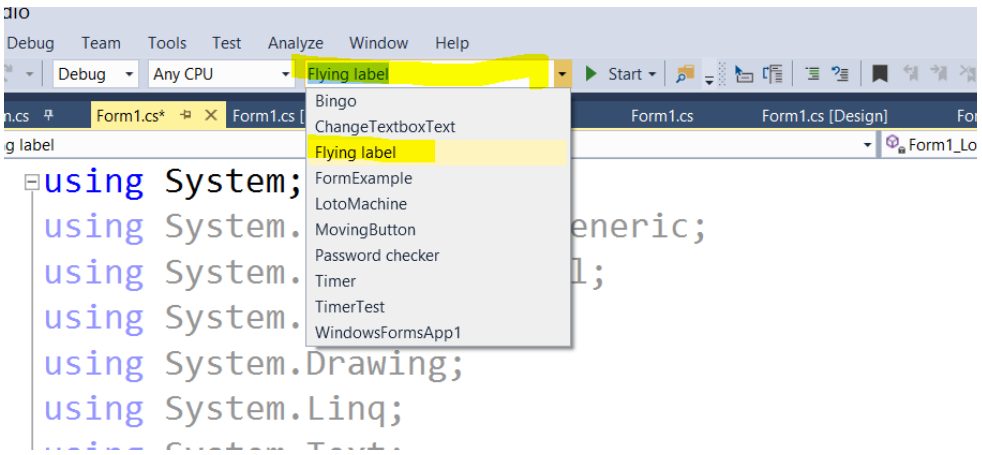
Real homework, submit these exercises!

NB! Submit all the exercises in one visual studio solution. Add different project for each task.

Right click on the solution and “Add new project.”



NB! To change the form that is displayed you have to choose it as a start up project:



```
using System;
using System.          eneric;
using System.          l;
using System.
using System.Drawing;
using System.Linq;
```

Projects and their content can be viewed in “SolutionExplorer” (view->solution explorer).

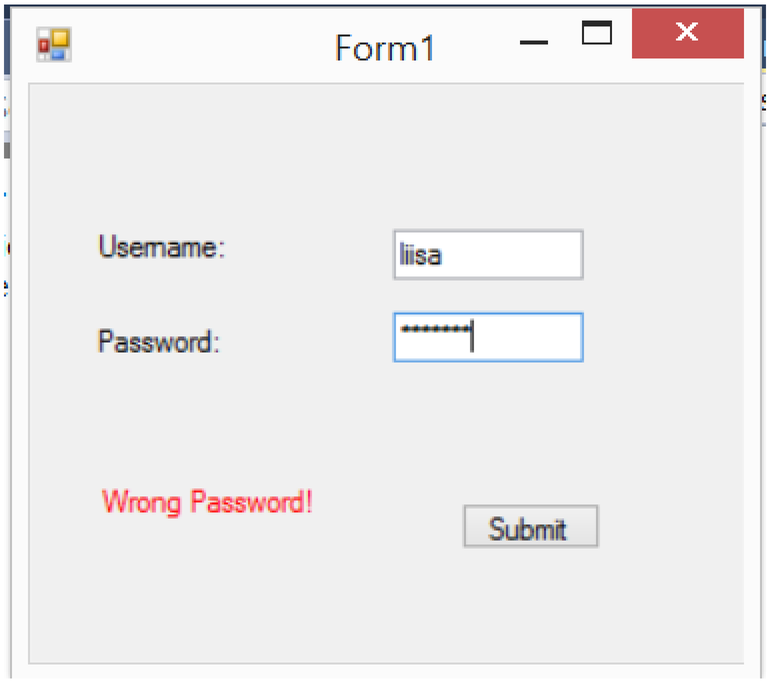
Homework: Exercise 1

Save a user name and password somewhere in the soltion (you can give values.)

Create a form for the user to enter a username and password. If the values are equal to the user name and password you saved before then display „Correct!“. If not then „Wrong password!“.

On the password field the password has to be replaced by „*“ characters. (Password text cannot be visible). The replacement has to happen after typing in every character.

The amount of * characters has to be equal to the user password. (If user entered a password with length 5 then ***** should be displayed).



Form1

Username:	liisa
Password:	********

Wrong Password!

Tips:

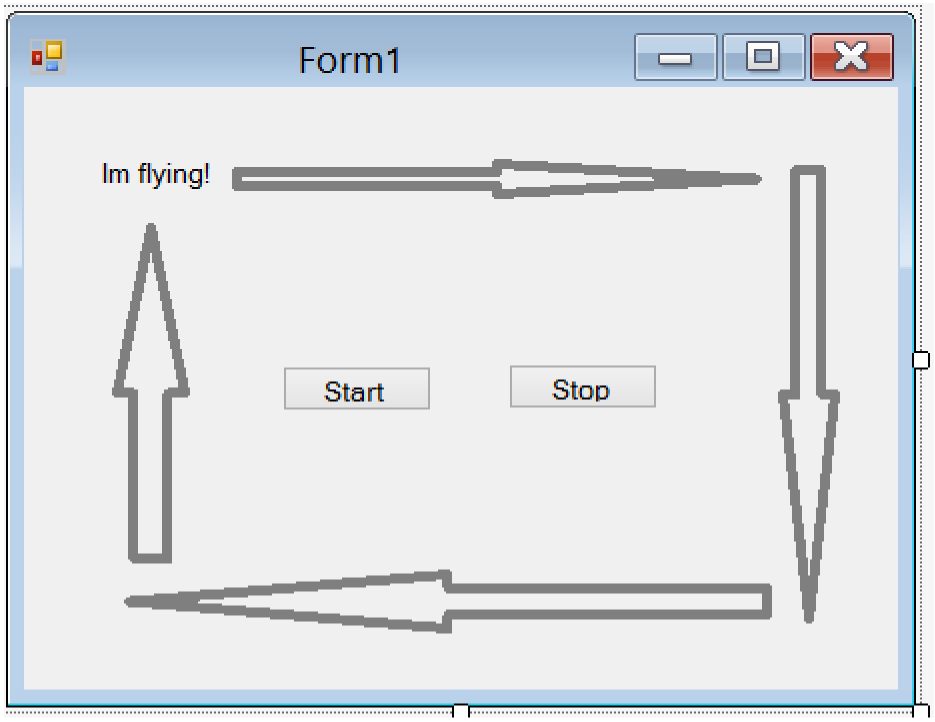
- Get it first working with visible password, replace the password with * sign as a last step. There are many ways to do it:
  - You have to save the text from the field every time a letter is entered
  - Get the password value by removing * signs from the final string when „submit“ is pressed
  - Get the entered letter every time when user enters a letter
  - Keep track of how many letters the user has entered!

Exercise 2: Flying label

Add a label to a form and change the text to „im flying“ and 2 buttons: Start and Stop.

When start button is pressed, make the label move around on the form (move the label after every 0.4 seconds). The approximate trajectory of the label is on the image below.

When stop button is pressed, the label should stop moving.



Tip: see MovingButton project from ained.ttu.ee.

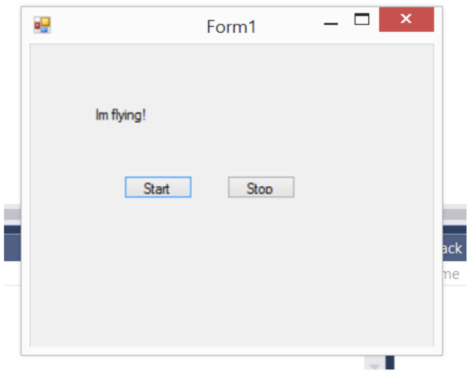
Draw a form on a paper and give test values to coordinates. Think of how they should change in order to get the label moving (when to increase and when to decrease).

Use labels for displaying coordinates to see how they change and what are the current values (*label1.Text = label1.Top*)

Top left coordinates are 0, 0.

You can position a label with your own coordinates to see how the positioning works (Example: *label1.Top=100, label1.Left =100*).

```
label1.Top = 50;
label1.Left = 50;
```

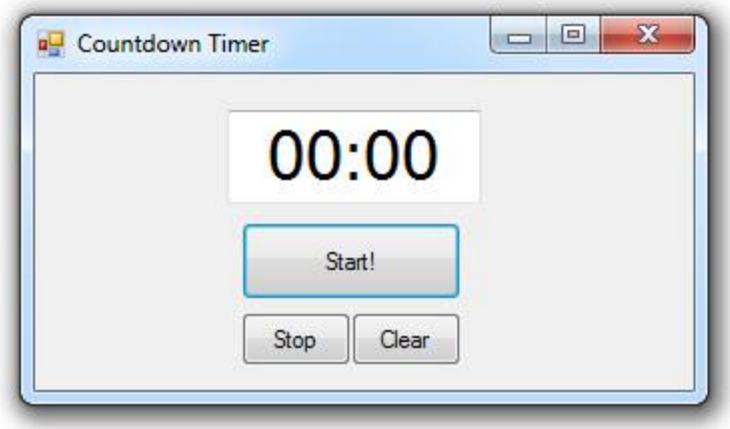


Exercise 3: countdown timer

Create a countdown timer where user can enter minutes and seconds to count down to 0. The timer is updated every second. When timer finishes (time is 00:00), the font turns to red.

If input string is in invalid format (not *mm:ss*) then a notification should be displayed. Timer starts only when the input is in a correct format.

Max value for seconds is 59 (00:65 is invalid input, 01:05 is valid)



Example: *03:06->03:05->03:04->03:02->03:01->03:00->02:59->02:58* … etc

Tips:

- "*00:00*" -> is just a string. The format is determined by ":" sign: minutes are on the left side and seconds on the right side. Remember: https://www.dotnetperls.com/split
- Test only with seconds first. If the timer works with seconds, then also add minutes.
- Our timer ticks correspond to real seconds, this can be used as helping information.
- If value of seconds is "00" and we have minutes left, then we should subtract 1 from minutes.