INF 154 LECTURE 3

- Planning a program?
- Designing the interface
- Organising interface elements
- The importance of program structure
- How to read and understand a program
- Useful tips

After this lecture you should …

1. Be able to analyse the problem to be solved and create an algorithm to implement the plan.
2. Be able to analyse the problem to be solved through a program, design the interface and plan the coding
3. Be able to read through an event handler and explain what it does (i.e. what input it uses, what processing happens and what the output will be when it is executed)
4. Know what to do when you have accidentally deleted an event handler or if you want to delete an unwanted, empty event handler.
5. Understand the importance of structuring your program well, using good variable names and comments
6. Become more comfortable using the programming environment and concepts learnt so far

Part 1: Planning Your Program

1. When given a **programming problem to solve**, you **cannot just jump in** and **start coding.**
2. You have to **think carefully about the steps** necessary **to solve** the **problem** (**the algorithm**) by thinking about
   a) What the **purpose of the program** is
   b) What **input is required** and **how best to get** that **input** from the user
   c) What **processing must be done** to solve the problem and **where (e.g. in what event handlers)** that processing should take place
   d) What **output should be presented** and **how best to present** it on the interface

3. A good way to **depict the algorithm** is by **drawing** up a **flow diagram** or **write pseudocode** of how the processing will proceed.

Planning a program

4. **Once you have an algorithm, you design the interface** – that is you have to think about how to lay out the forms, what controls must be placed on the forms and how best to organise them so as to make the program as easy to use as possible.

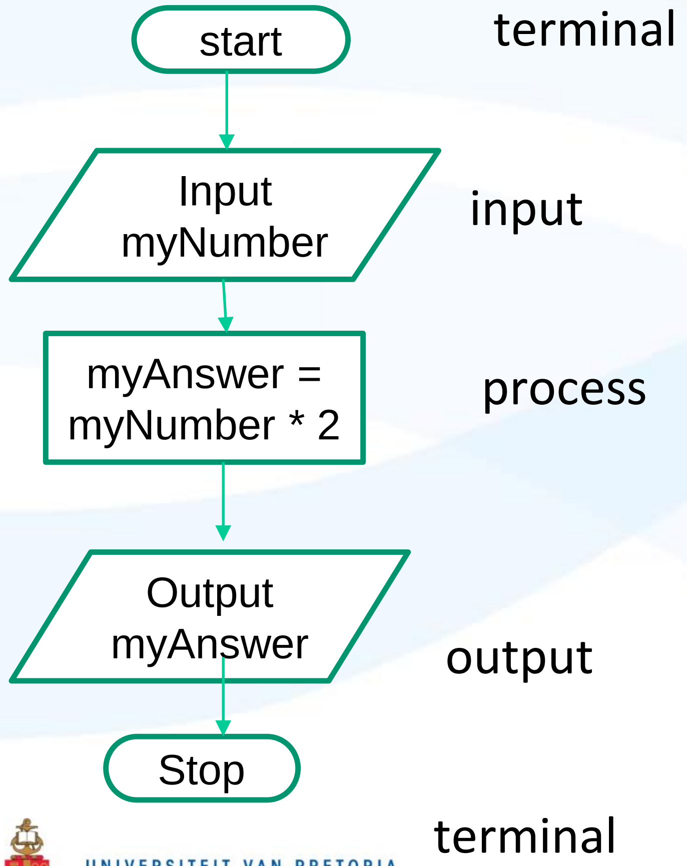
5. **Now you can start thinking about how to write code** for the different event handlers (i.e. to implement the processing identified in step 2c) above.

6. **The flow diagram can then be translated into program code.** Include comments in the program to explain to a reader what is happening.

7. **The last and important step is to test the program with all** possible forms of input to make sure it works correctly for all cases.

Developing an algorithm - example

Give the flow chart and pseudocode for a program that doubles a number



```
start
  input myNumber
  myAnswer = myNumber*2
  output myAnswer
stop
```

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Denkleiers • Leading Minds • Dikgopolo tša Dihlalefi

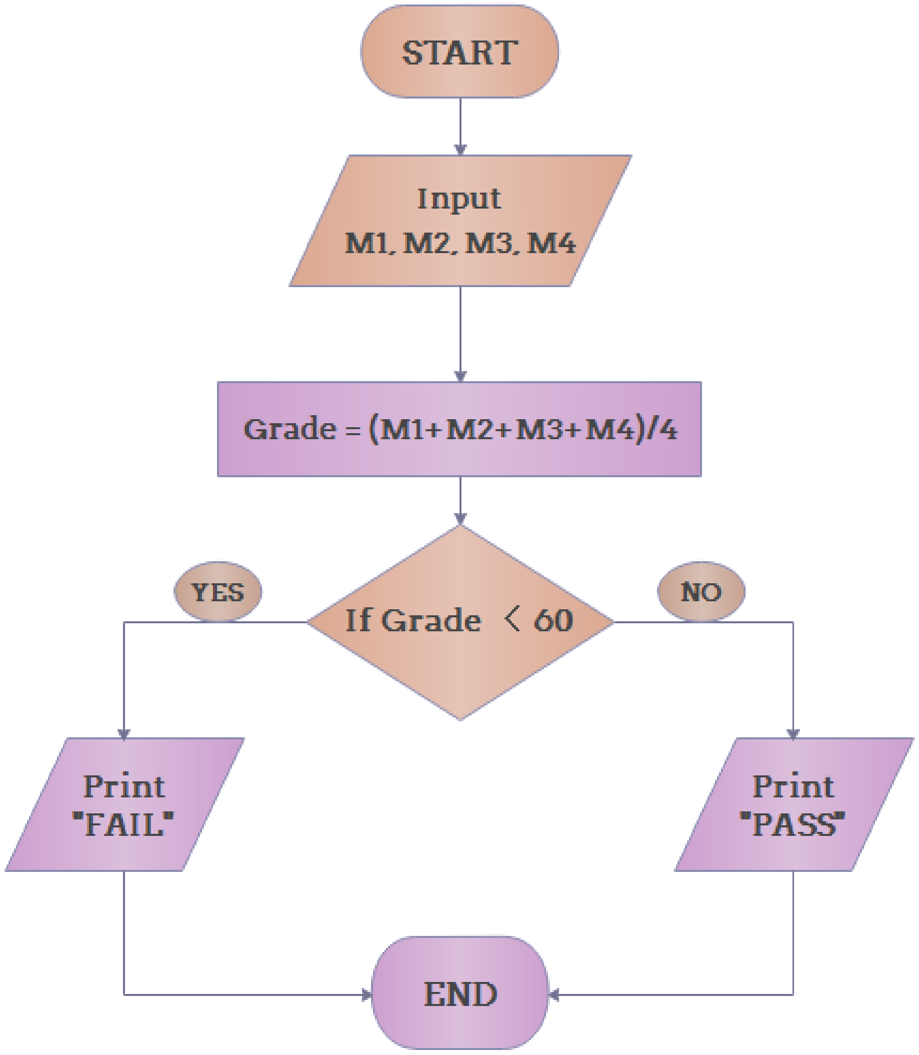
Developing an algorithm – more examples (flow charts)

1. Algorithm for a program that calculates if a student passed / failed?

START

- Step 1: Input grades of 4 courses M1, M2, M3 and M4,
- Step 2: Calculate the average grade with formula "Grade=(M1+M2+M3+M4)/4"
- Step 3: If the average grade is less than 60, print "FAIL", else print "PASS".

STOP



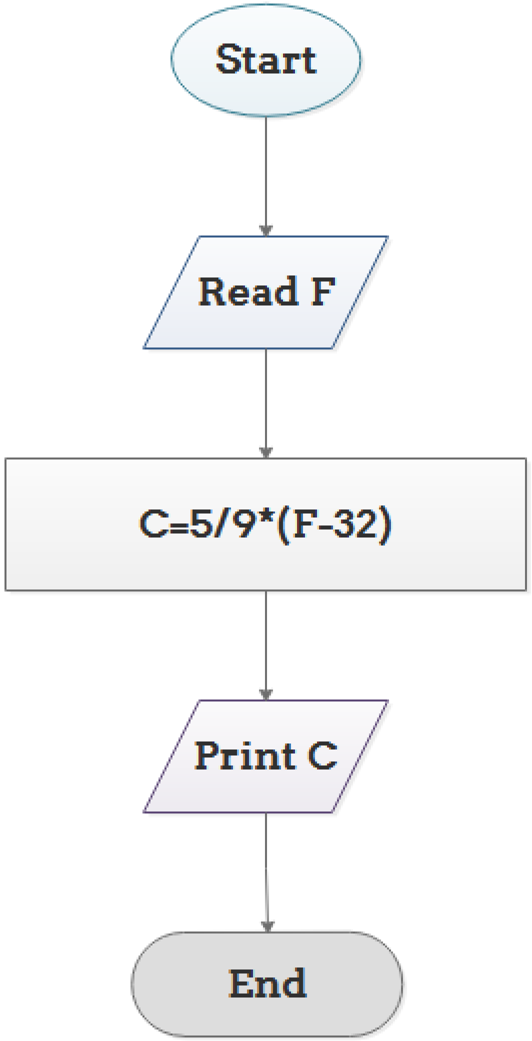
Developing an algorithm – more examples (flow charts)

1. Algorithm for a program that converts Temperature from Fahrenheit (°F) to Celsius (°C)

START

- Step 1: Read temperature in Fahrenheit,
- Step 2: Calculate temperature with formula C=5/9*(F-32),
- Step 3: Print/Display C,

STOP



UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Denkleiers • Leading Minds • Dikgopolo tša Dihlalefi

Planning a program (Exercise 1)

Consider the following programming problem:

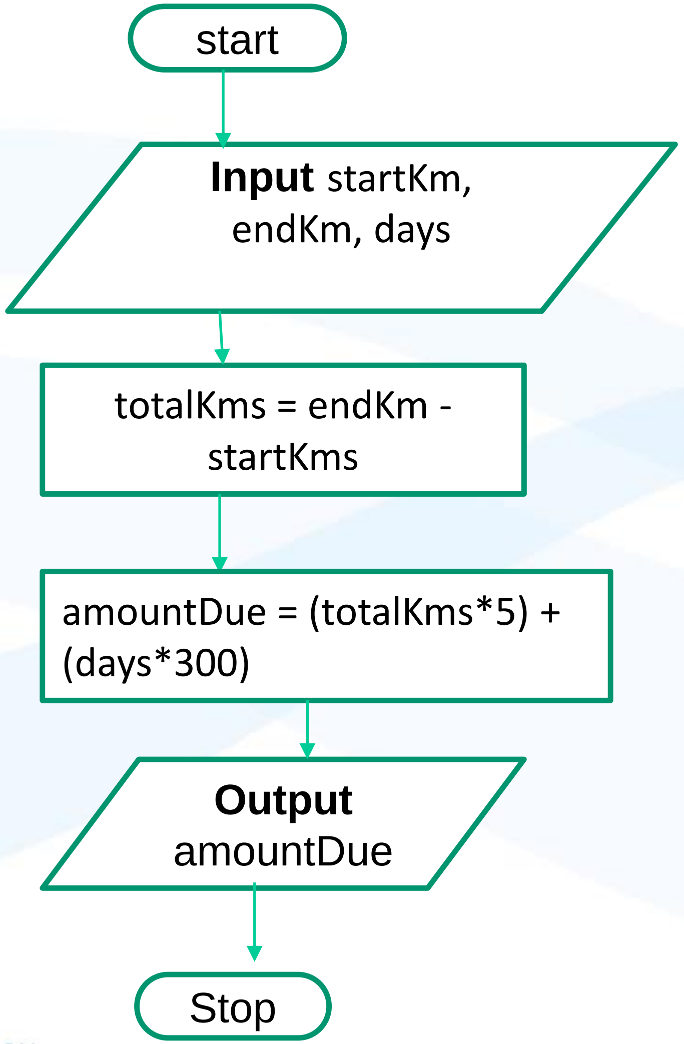
1.Create a program for a car rental agency to **calculate car rental charges**. They charge **R300.00 per day** and **R5.00 per km**.

2.When a customer returns a car, the **user will enter** the **beginning** and **end odometer reading** (in kilometres) and the **number of days** the car was rented.

3.The program must calculate and display the amount owed by the customer.

Formula for rental charge:
amountDue = (totalKmsTravelled*5) + (days*300)

Planning a program (Exercise 1) – algorithm (flowchart)


start
Input startKm, endKm, days
totalKms = endKm - startKms
amountDue = (totalKms*5) + (days*300)
Output amountDue
Stop

Planning a program

A)What is the purpose of the program?

To calculate the amount owed by a car rental agency customer when a car is returned.

B)What input is required?

1. Beginning and end odometer reading in kilometres (no decimal values required)
2. Number of days the car was rented (part of a day counts as a full day, so this will be an integer as well)

C)What is the best way to get this input?

Two text boxes (for the kilometres) and a numeric updown (for the days) accompanied by labels that indicate to the users what should be entered there.

Planning a program

C)What processing must be done to solve the problem?

1. The three input values should be taken from the text boxes, converted into integers and stored in suitably named variables
2. The total kilometres travelled must be calculated by subtracting the beginning km reading from the end reading.
3. The total kms travelled must be multiplied by 5.
4. The number of days must be multiplied by R300.
5. The answers to 2 and 3 must be added together.

D)Where should this processing take place?

There should be a “Calculate price” button. When the user clicks on it, it should do the calculation. (So, in a button’s Click event handler.)

Planning a program

E)What output should be presented?

The total amount due by the customer must be displayed.

F)How should we present it on the interface?

It should appear with a message “The amount due is: “ followed by the amount. The amount must be preceded by “R”.

This output string can either be presented on the interface in a Label control or it can be displayed in a MessageBox.

Now that we have a clear understanding of what the program entails and how it can be implemented, we can design the interface. You have to think about how to lay out the form, what controls must be placed on the form and how best to organise them. Also think how you can make it easy on the user.

Planning a program

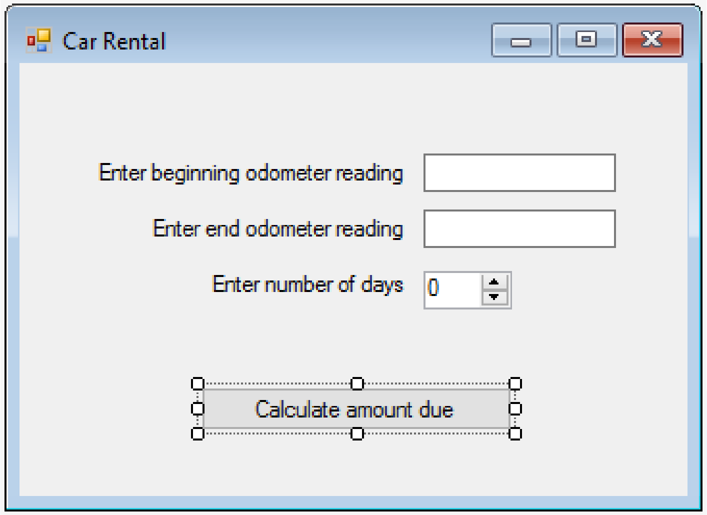
Plan your Screen: To design the interface we have to determine which controls should be added and how their property values should be set.

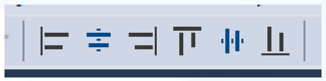
We will have one form with the following controls/properties:

Control	Property	Value
Form	Name	frmCarRental
	Text	Car Rental
Label	Text	Enter beginning odometer reading
TextBox	Name	txtStartOdoReading
Label	Text	Enter end odometer reading
TextBox	Name	txtEndOdoReading
Label	Text	Enter number of days
NumericUpDown	Name	nudDays
Button	Name	btnCalculate
	Text	Calculate amount due

Planning a program

The Form design:



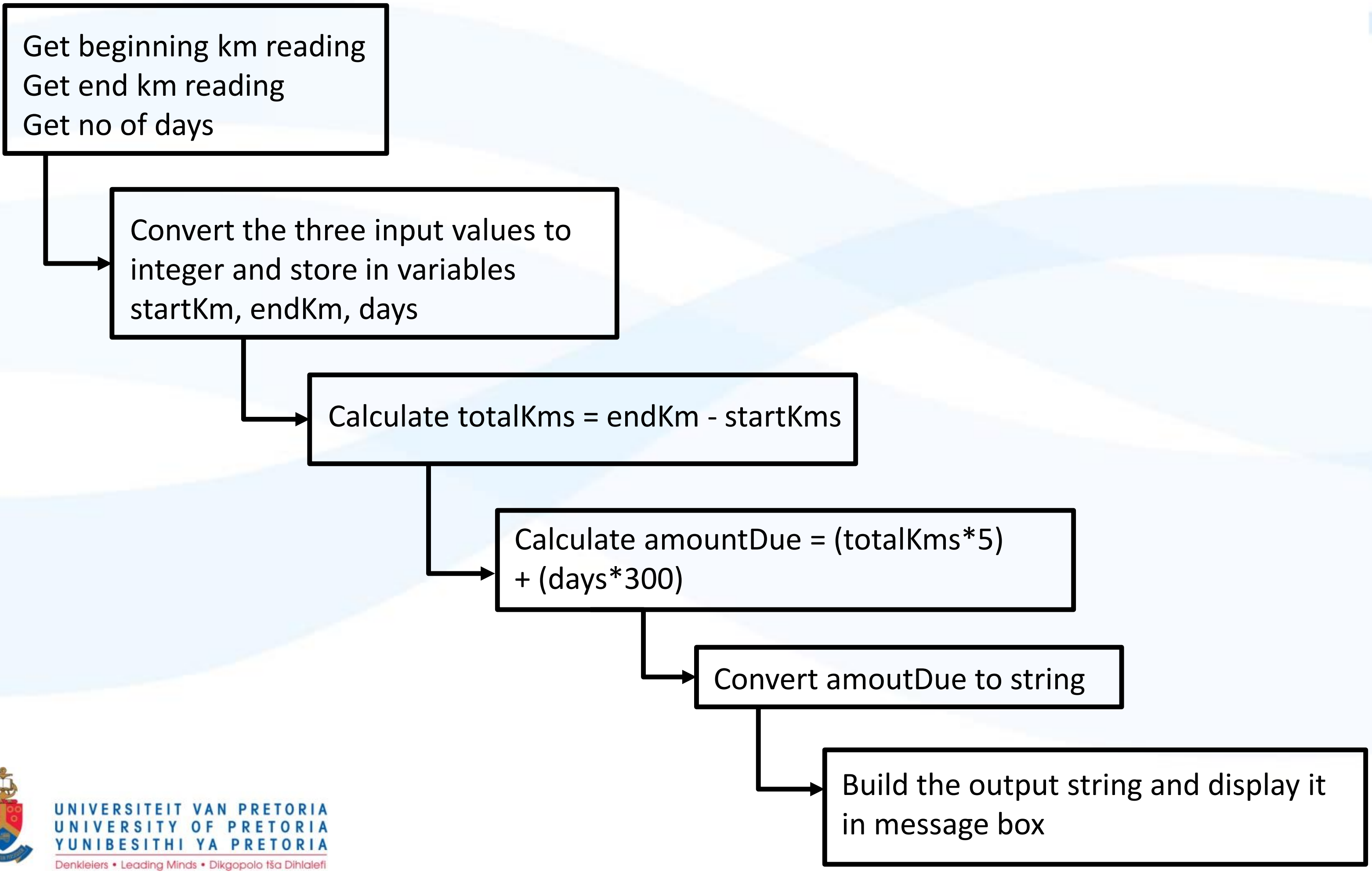
<u>Make sure the layout looks neat</u> – use these buttons  in the menu to left-align, right-align, etc. a group of selected controls.

You can also set the TabIndex properties of the three input controls and the button to 0, 1, 2 and 3 respectively, so that the user can move from one to the other with the Tab key in a logical order. The control with TabIndex 0 will be the active control when the form comes up.

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Denkleiers • Leading Minds • Dikgopolo tša Dihlalefi

Planning a program

Now we have to create the Click event handler for the button to do the processing. But let's think about the flow of the processing that will happen here first:



Get beginning km reading
Get end km reading
Get no of days

Convert the three input values to integer and store in variables startKm, endKm, days

Calculate totalKms = endKm - startKms

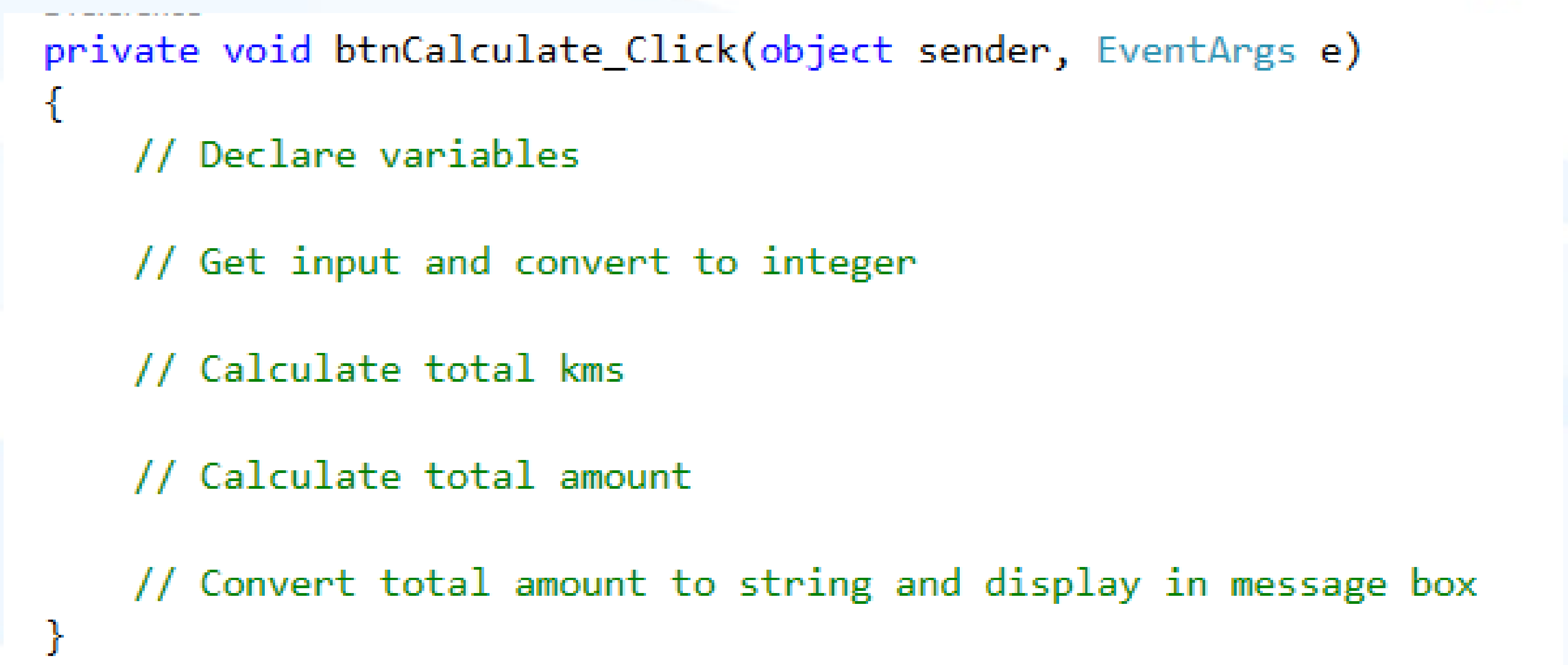
Calculate amountDue = (totalKms*5) + (days*300)

Convert amoutDue to string

Build the output string and display it in message box

Planning a program

We can **translate those steps into comments** in the Click event handler as follows. Then we are ready to write the program statements.



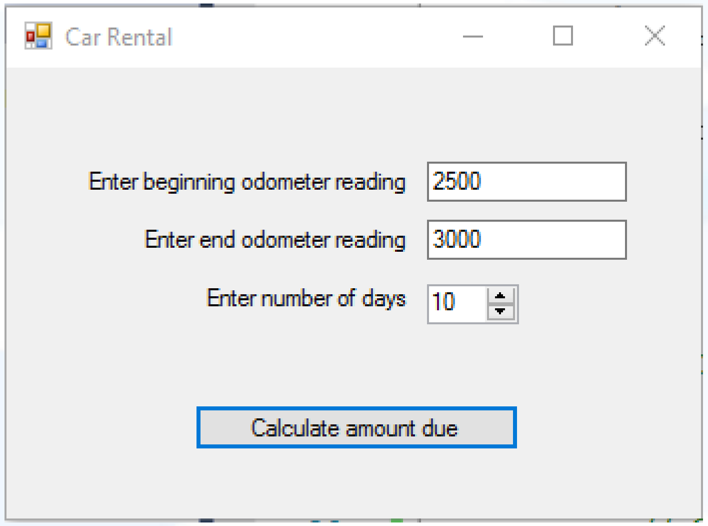
```
private void btnCalculate_Click(object sender, EventArgs e)
{
    // Declare variables

    // Get input and convert to integer

    // Calculate total kms

    // Calculate total amount

    // Convert total amount to string and display in message box
}
```

Planning a program

```
private void btnCalculate_Click(object sender, EventArgs e)
{
    // Declare variables
    int startKms, endKms, days;
    int totalKms, totalAmount;

    // Get input from TextBoxes and NUD and convert to integer
    startKms = Convert.ToInt32(txtBeginningOdodReading.Text);
    endKms = Convert.ToInt32(txtEndOdoReading.Text);
    days = Convert.ToInt32(nudDays.Value);

    // Calculate total kms
    totalKms = endKms - startKms;

    // Calculate total amount
    totalAmount = (totalKms * 5) + (days * 300);

    // Convert total amount to string and display in message box
    MessageBox.Show("The amount due is R" +
                                   Convert.ToString(totalAmount));
}
```

Planning a program

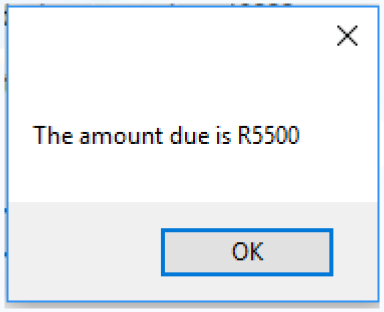
Finally we can **test the program.**



Enter beginning odometer reading 2500

Enter end odometer reading 3000

Enter number of days 10


The amount due is R5500

Part 2 – How to Read and Understand Code

Variables

Constants

Comments

Structure of Code

Interface

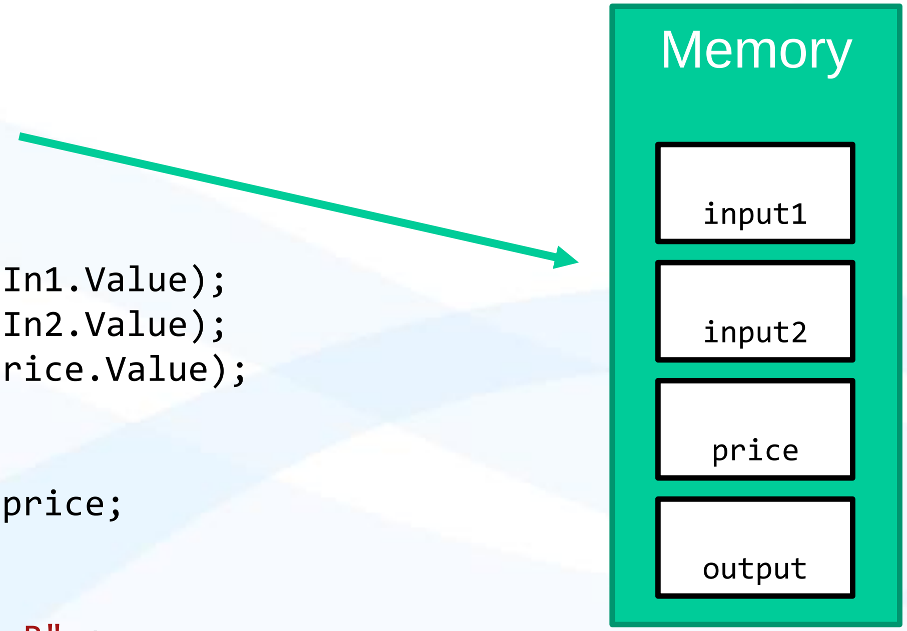
How to read and understand program code (Exercise 2)

You must also be able to read the code of an existing program and figure out what it does. Consider the following code. Can you determine what it does and what the output will be if the user types the value 10 into all three numeric updown controls?

```
private void btnCalculate_Click(object sender, EventArgs e)
{
    // Declare variables
    double input1, input2, price;
    double output;

    // Get input
    input1 = Convert.ToDouble(nudIn1.Value);
    input2 = Convert.ToDouble(nudIn2.Value);
    price = Convert.ToDouble(nudPrice.Value);

    // Do calculation
    output = (input1 * input2) * price;

    // Display result
    MessageBox.Show("The cost is: R" + Convert.ToString(output));
}
```

How to read and understand program code

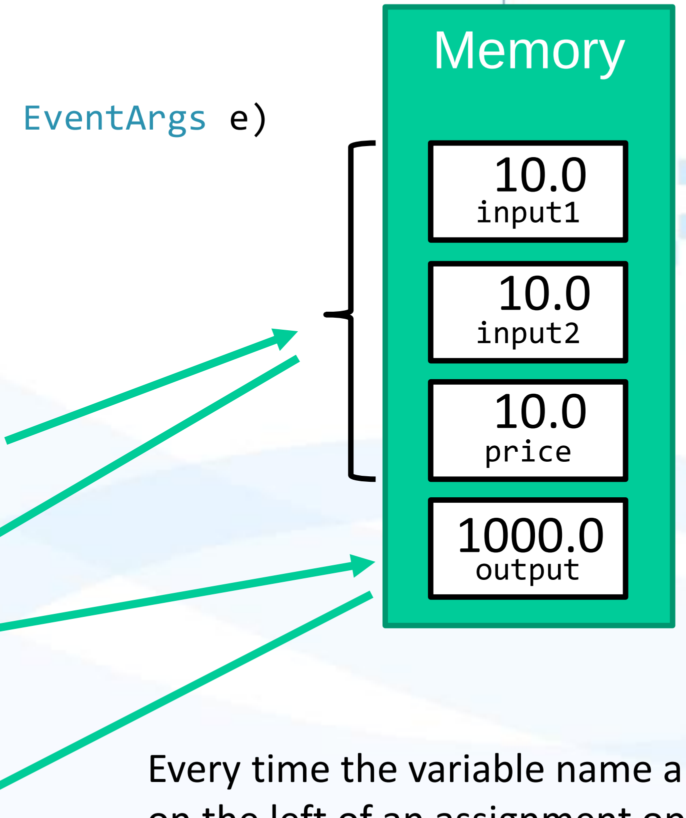
```
private void btnCalculate_Click(object sender, EventArgs e)
{
    // Declare variables
    double input1, input2, price;
    double output;

    // Get input
    input1 = Convert.ToDouble(nudIn1.Value);
    input2 = Convert.ToDouble(nudIn2.Value);
    price = Convert.ToDouble(nudPrice.Value);

    // Do calculation
    output = (input1 * input2) * price;

    // Display result
    MessageBox.Show("The cost is: R" +
                          Convert.ToString(output));
}
```



Think of the computer's memory as a cabinet that gets an empty drawer for each variable declaration.

How to read and understand program code

```
private void btnCalculate_Click(object sender, EventArgs e)
{
    // Declare variables
    double input1, input2, price;
    double output;

    // Get input
    input1 = Convert.ToDouble(nudIn1.Value);
    input2 = Convert.ToDouble(nudIn2.Value);
    price = Convert.ToDouble(nudPrice.Value);

    // Do calculation
    output = (input1 * input2) * price;

    // Display result
    MessageBox.Show("The cost is: R" +
                          Convert.ToString(output));
}
```



Every time the variable name appears on the left of an assignment operator, a value is placed in its drawer.

Every time it appears elsewhere (e.g. on the right of an assignment operator) the value is taken from the drawer in memory.

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Denkleiers • Leading Minds • Dikgopolo tša Dihlalefi

How to read and understand program code

```
private void btnCalculate_Click(object sender, EventArgs e)
{
    // Declare variables
    double input1, input2, price;
    double output;

    // Get input
    input1 = Convert.ToDouble(nudIn1.Value);
    input2 = Convert.ToDouble(nudIn2.Value);
    price = Convert.ToDouble(nudPrice.Value);

    // Do calculation
    output = (input1 * input2) * price;

    // Display result
    MessageBox.Show("The cost is: R" +
                          Convert.ToString(output));
}
```

<u>So what does this program do?</u>
It gets three input values from the user – one of which is a price.
It multiplies the three values.
So, if the values are 10.0, 10.0 and 10.0, the output will be: “The cost is R 1000.0”.

We can thus work out the output value based on the input provided, but its is not 100% clear who will use this program.

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Denkleiers • Leading Minds • Dikgopolo tša Dihlalefi

How to read and understand program code

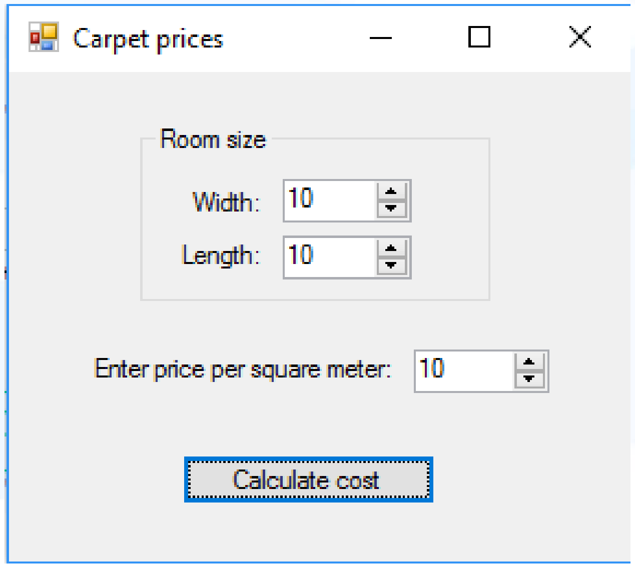
The variable names used in the two previous slides are very generic and almost meaningless. If we use meaningful variables it is much easier to read what the program does.

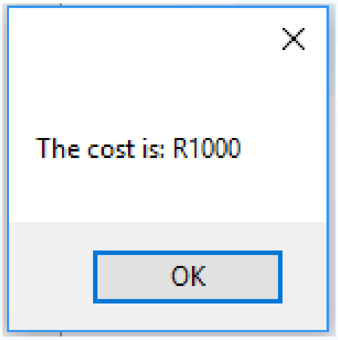
```
private void btnCalculate_Click(object sender, EventArgs e)
{
    // Declare variables
    double width, length, pricePerSqMeter;
    double cost;

    // Get input
    width = Convert.ToDouble(nudWidth.Value);
    length = Convert.ToDouble(nudLength.Value);
    pricePerSqMeter = Convert.ToDouble(nudPricePerM.Value);

    // Do calculation
    cost = (width * length) * pricePerSqMeter;

    // Display result
    MessageBox.Show("The cost is: R" + Convert.ToString(cost));
}
```

How to read and understand program code

If we were given the interface, it would have been very easy to determine what the program does:

It asks the user for the size of a room and the price of a carpet per square meter. Then calculates the cost of the carpet.




The cost is: R1000

How to read and understand program code (Exercise 3)

Let’s work out what the code below does. Will it work correctly?

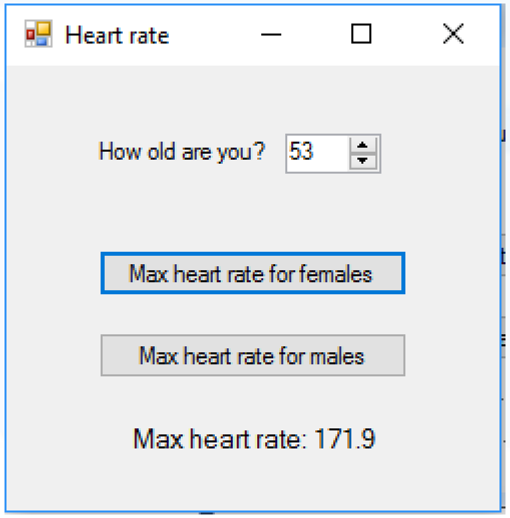
```
private void btnFemaleHeartRate_Click(object sender, EventArgs e)
{
    int age;
    int result;

    const double X = 0.7;

    age = Convert.ToInt32(nudAge.Text);
    result = 209 - (X * age);

    lblResult.Text = "The result is: " + result;
}
```

Not only should you be able to work out what output a program will give, you should also be able to find the errors without the help of Visual Studio.

How to read and understand program code

A better solution:

```
private void btnFemaleHeartRate_Click(object sender, EventArgs e)
{
    // Declare variables
    int age;
    double maxHeartRate;
    const double X = 0.7; // value for females

    // Get user's age
    age = Convert.ToInt32(nudAge.Value);

    // Calculate max heart rate for females
    maxHeartRate = 209 - (X * age);

    // Display the result
    lblResult.Text = "Max heart rate: " +
                     Convert.ToString(maxHeartRate);
}
```



Max heart rate: 171.9

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Denkleiers • Leading Minds • Dikgopolo tša Dihlalefi

How to read and understand program code

The event handler for males looks almost the same, but the fomula that we have to implement here is:

Maximum heart rate = 214 – (0.8 x age)

```
private void btnMaleHeartRate_Click(object sender, EventArgs e)
{
    // Declare variables
    int age;
    double maxHeartRate;
    const double X = 0.8; // value for males

    // Get user's age
    age = Convert.ToInt32(nudAge.Value);

    // Calculate max heart rate for females
    maxHeartRate = 214 - (X * age);

    // Display the result
    lblResult.Text = "Max heart rate: " +
                      Convert.ToString(maxHeartRate);
}
```

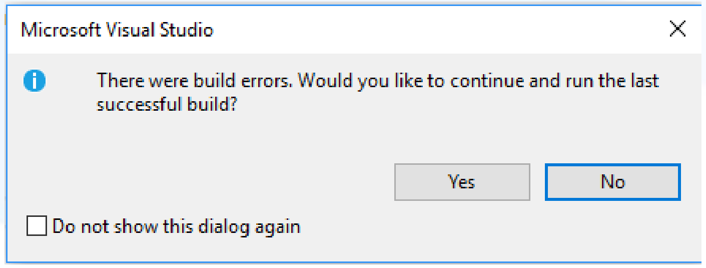
Part 3– Getting rid of unwanted code

By now it must have happened to you a few times that you accidentally double-clicked on a control and it opened up an event handler that you did not really want.

For example, when writing the heart rate program, I double-clicked on the result label and it opened up the event handler below:

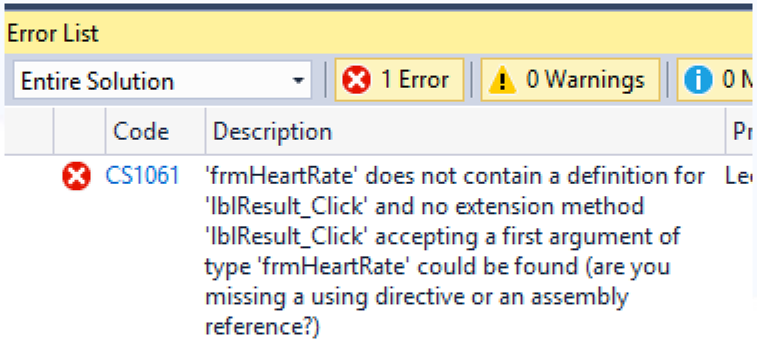
```
private void lblResult_Click(object sender, EventArgs e)
{

}
```

Since I don't want any processing done when the user clicks on the output I want to remove this event handler, but I CANNOT JUST DELETE IT. The following will appear:



Microsoft Visual Studio

There were build errors. Would you like to continue and run the last successful build?



Error List

Code	Description	Pr
CS1061	'frmHeartRate' does not contain a definition for 'lblResult_Click' and no extension method 'lblResult_Click' accepting a first argument of type 'frmHeartRate' could be found (are you missing a using directive or an assembly reference?)	Le

Getting rid of unwanted code

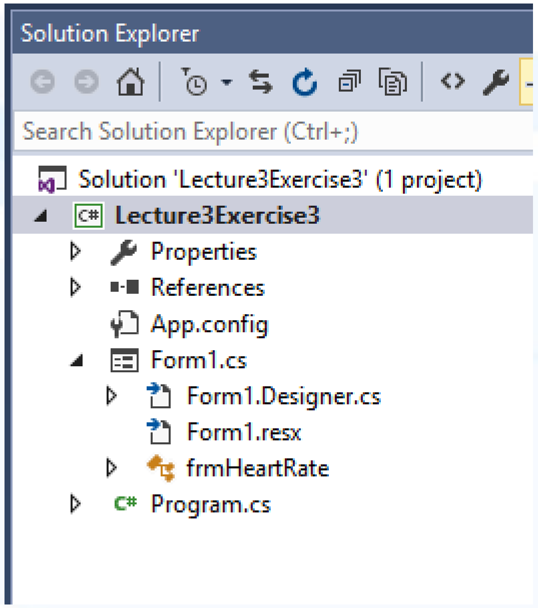
You should never just delete code that was generated by Visual Studio.

WHY?

A C# project consists of a number of components. If you look in the Solution Explorer window, you will see a whole list of files that make up your project.

When Visual Studio adds code to your Form it also adds code to some of these other program files, so if you just delete something from your form, that code will still be in the other files and your program will not work.

It is, however, annoying to have to keep unwanted, empty event handlers in your code, so we will explain how to go about removing them from your project.



Solution Explorer

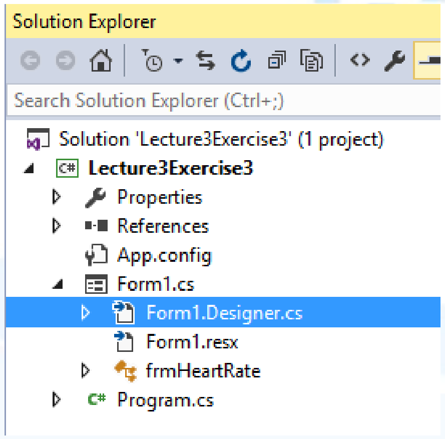
Search Solution Explorer (Ctrl+;)

```
Solution 'Lecture3Exercise3' (1 project)
  Lecture3Exercise3
    Properties
    References
    App.config
    Form1.cs
      Form1.Designer.cs
      Form1.resx
      frmHeartRate
    Program.cs
```

Getting rid of unwanted code

Suppose I want to delete the empty lblResult_Click event handler from my project. Here is what I do.

In the form's code, delete the whole event handler.

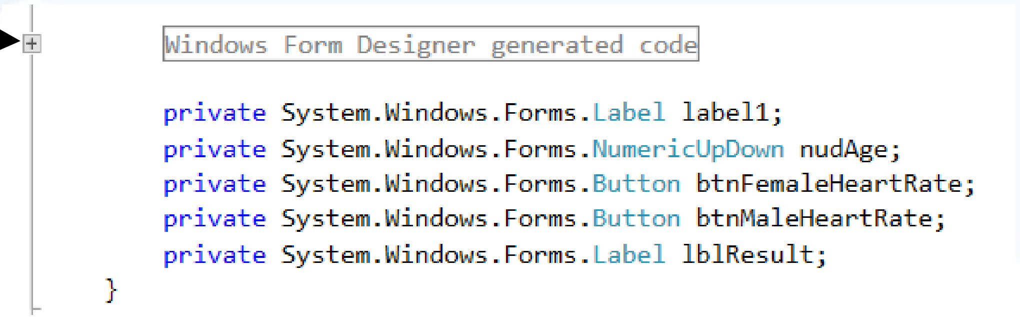


In the Solution Explorer, double-click on Form1.Designer.cs

In the editing pane, find the following code:

Click on the +.

This will open up more lines of code.

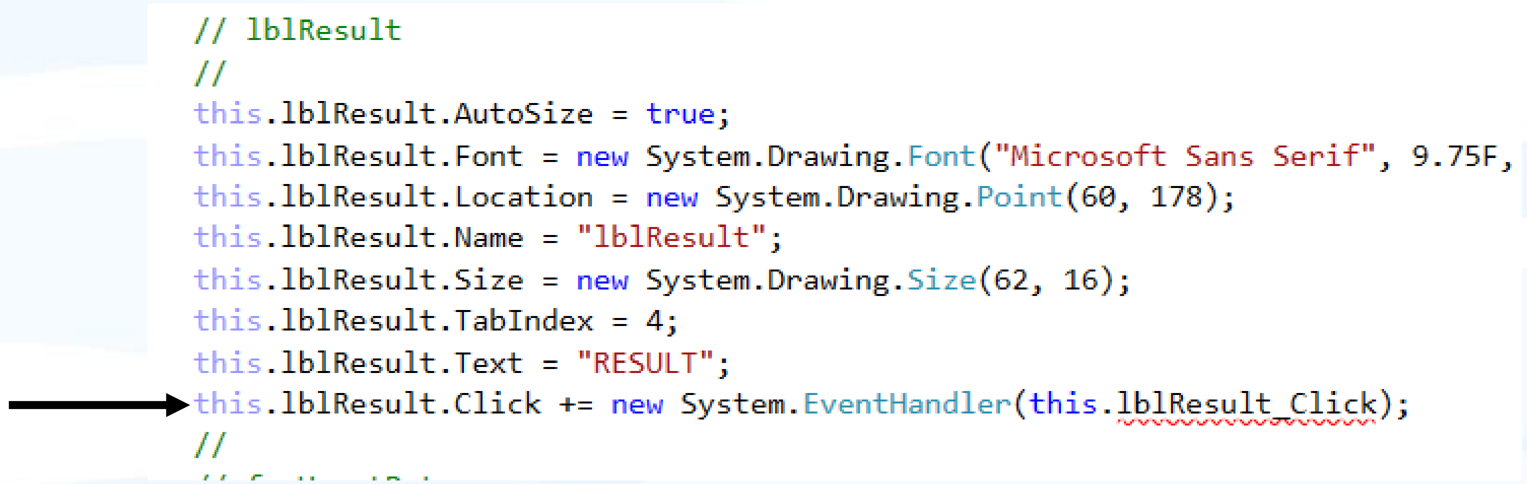


```
        Windows Form Designer generated code

        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.NumericUpDown nudAge;
        private System.Windows.Forms.Button btnFemaleHeartRate;
        private System.Windows.Forms.Button btnMaleHeartRate;
        private System.Windows.Forms.Label lblResult;
    }
```

UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA
Denkleiers • Leading Minds • Dikgopolo tša Dihlalefi

Getting rid of unwanted code

Scroll down through the code until you find the code where the Click event handler for lblResult is created and delete that whole line.



```
// lblResult
//
this.lblResult.AutoSize = true;
this.lblResult.Font = new System.Drawing.Font("Microsoft Sans Serif", 9.75F,
this.lblResult.Location = new System.Drawing.Point(60, 178);
this.lblResult.Name = "lblResult";
this.lblResult.Size = new System.Drawing.Size(62, 16);
this.lblResult.TabIndex = 4;
this.lblResult.Text = "RESULT";
this.lblResult.Click += new System.EventHandler(this.lblResult_Click);
//
```

You should now be able to run the program again.