Revision Labs

## Objective

In these series of lab you will

* Create a program to practice many of the pre-course studies

## Part 1 – Make a start!

Visual Studio is the development tool used by many C# developers and throughout this course. This lab will show you how to create and run a C# program.

## Duration

## 5 minutes

## Step by step

### Creating a new Console application using Visual Studio 2022

1. Launch Visual Studio and then create a new project Text

   Description automatically generated
2. Select Console app Logo

   Description automatically generated with low confidence and then Click **Next**
3. Type **C:\Labs** as the location for your project and type **Labs** for the name
4. You may choose this option if you would like to keep your project and solution file together, but it has little consequence 
5. Click Next
6. Select for this session so our project layout matches the previous versions of .NET
7. Click **Create**
8. **A starter code is created for you like**

namespace Labs

{

internal class Program

{

static void Main(string[] args)

{

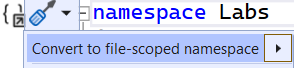
Console.WriteLine("Hello, World!");

}

}

}

1. For a cleaner layout and to reduce the number of brackets, hover over the {} icon and then select the first option

  
Alternatively, you can change the code manually

namespace Labs;

internal class Program

{

static void Main(string[] args)

{

Console.WriteLine("Hello, World!");

}

}

### **Building and running the program**

1. Press **Ctrl+F11** to run the program or click the green **Run** icon.

**\*\* End of part 1 \*\***

### **Part 2 - Creating methods**

1. Add a new method to the Program class and just under the Main method as **public** **static** **int** **GetInt**(**String** prompt)

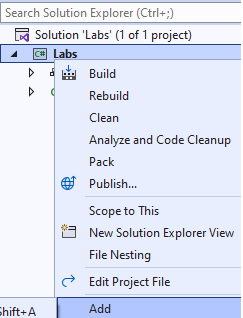
This method displays a prompt (String prompt) and then gets an integer input (using the **Console.ReadLine** method) from the user and returns it as an **int**. (tip: use **int.Parse** method)

1. Create another static method called **string** **GetString**(**string** prompt)  
   This method displays a prompt (string prompt) and then gets a string from the user and then returns it as a **srting**.
2. Create another method called **Lab1()**
3. Call the **Lab1()** methodin the **Main()** method
4. Call both methods in the **Lab1()** method to:
   1. Get the user’s age
   2. Get the user’s age
   3. Display a message that includes the name and age+1 like:  
      (if Bob is 20 years old)  
        
      Dear **Bob** next year you will be **21** years old.

## Move your code to a separate class

Does every method have to be in the Program class?  
In this part you'll create a new class called Utils and then move the **GetInt** abd **GetString** methods to the Utils class.

1. Create a new Class called **Utils** .

**Tip:** Right mouse click on your project and then select **Add > Class** menus  
  


1. Transfer the code for the **GetInt** and **GetString** methods to the ***Utils*** class.
2. Remove the **static** word from every method definition.   
   We'll discuss **static** method later. The only reason why every method was static was because the **Main()** method in the **Program** class was a **static** method, but we are now free of static main()!
3. At the start of the **Lab1**() method, create an instance of the ***Utils*** class

***Utils*** **utils** = **new** ***Utils***();

1. At the start of each method call add "**utils**". For example:  
   instead of **GetInt()** type **utils.GetInt()**
2. Run the application to make sure everything works.

**\*\* End of Part 2 \*\***

## Part 3 - Practice IF statements

In this part you will calculate the amount of Life insurance

1. Comment out the code in part 1 & 2.
2. Create a new Class called **LifeInsurance.**
3. Create a new method in the LifeInsurance class called **Calculate** with no parameters which returns a double like public double Calculate()
4. Your **Lab1()** method will call the calculate method like:  
   (please comment out any previous code)

LifeInsurance life = new LifeInsurance(); Console.WriteLine(life.Calculate());

1. Calculating life insurance depends on many factors but for simplicity do:
   1. Get the customer’s **name** and **age**  
      We only consider ages between 18 and 7 5 (inclusive).   
      Please show a suitable message if age is not within the range and return **-1**.
   2. Get the number of years the user wishes to keep the life insurance running.   
      The duration must be between **10 and 25** (inclusive)  
      Older people may not be able to get the terms they ask for. The maximum number of years is calculated as **85 – age.**  
      Therefore, a 70 year old person may only run the scheme for 15 years (**70+15=85)**.
   3. Ask the user for their occupation.  
      Apply a **10%** discount for nurses, doctors, teachers and police officers.
   4. Apply a formula like:  
      Minimum payment of **£5 + Age + (Term/10)**  
      Apply any discount and then return the result.

**\*\*\* End part 3 \*\*\***

## Part 4 - Practice an Iteration statement (loops)

In this part you will calculate the amount of Life insurance

1. Create a **for loop** in the **Lab1()** method, execute the line containing life.Calculate(); **3** times.
2. Replace the **for loop** with a **while loop**.
3. Test you code to make sure it works
4. Within the while loop ask the user if they would like to calculate another quote and break out of the loop if they say No!

**\*\*\* End of the Revision exercises \*\*\***