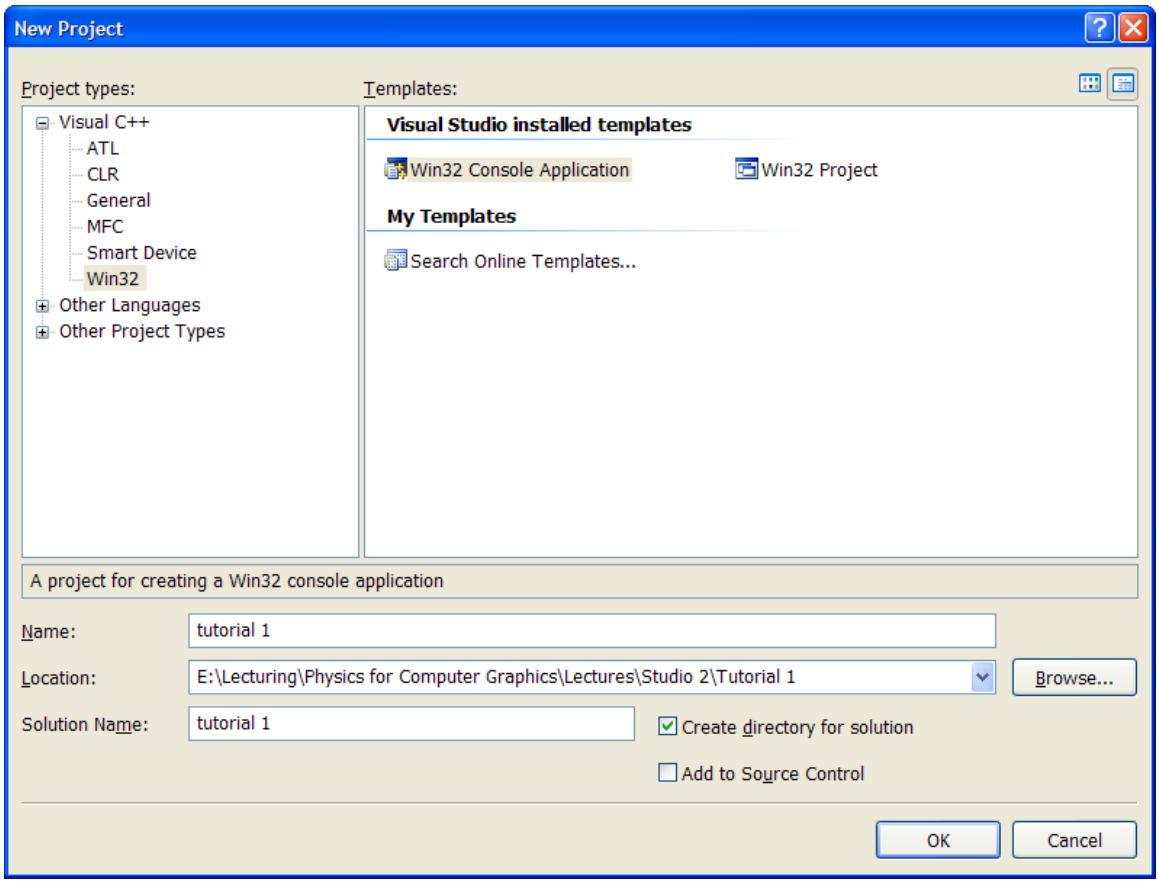
Introduction to C++ (Part I)

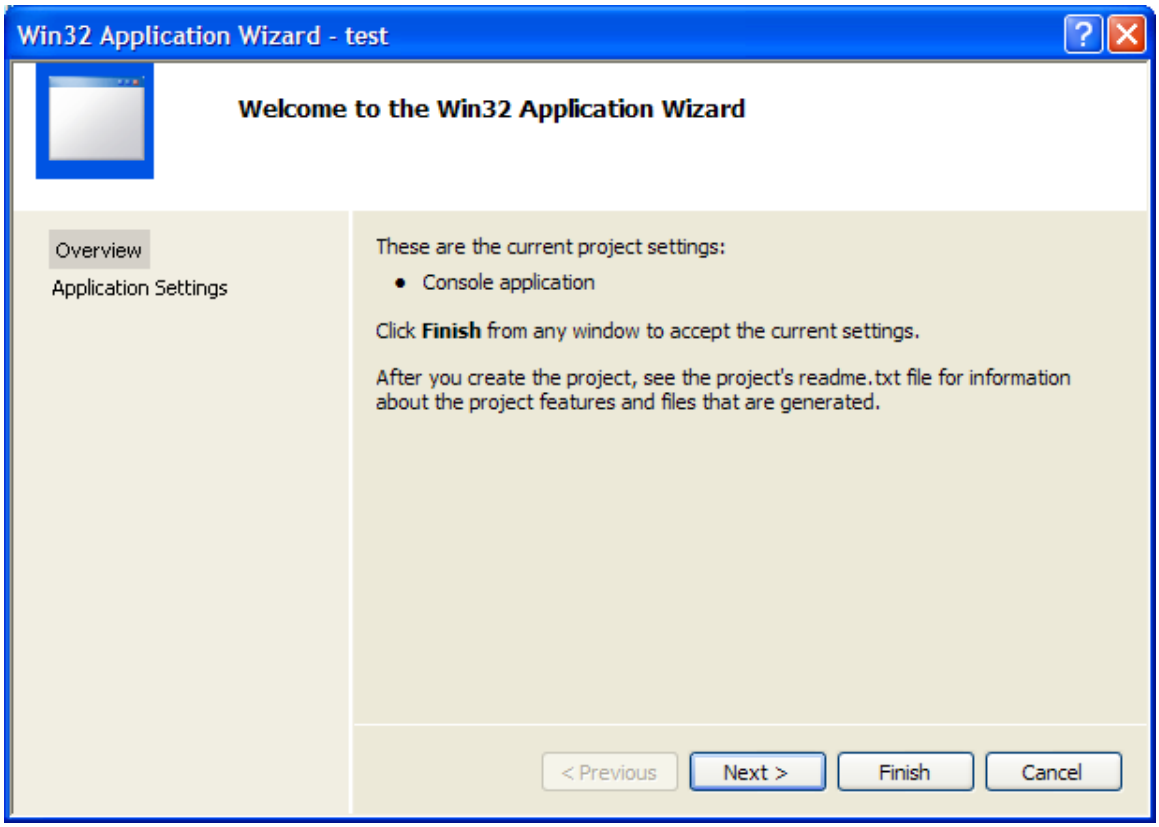
Welcome to the first laboratory session which will introduce you to C++.

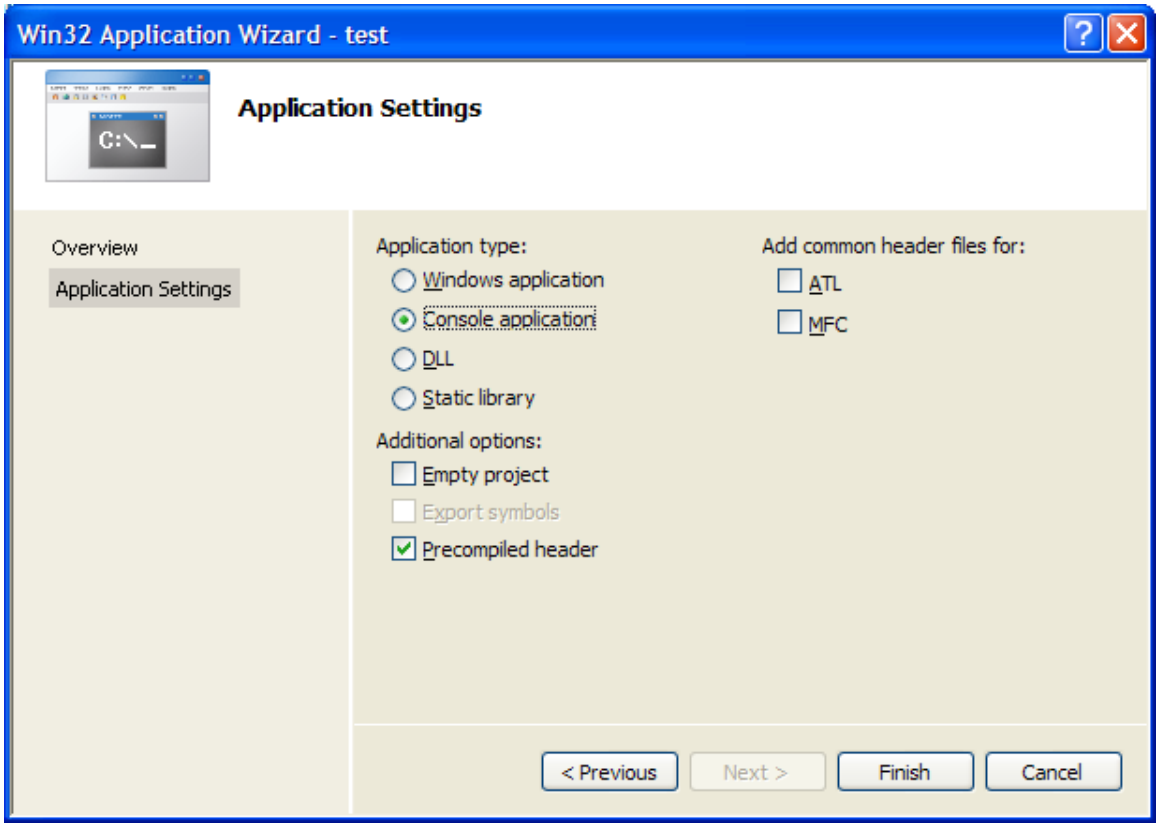
Setting Up

First start Microsoft Visual C++. Then go to *File->New Project*. This will open the project window. Then select Win32->Win32 Console Application, and enter the directory that you want to store the source code as well as the name of source file. Select as *Tutorial 1* for the name of the directory and as *tutorial 1* for the name of the source file as shown from the figure below.



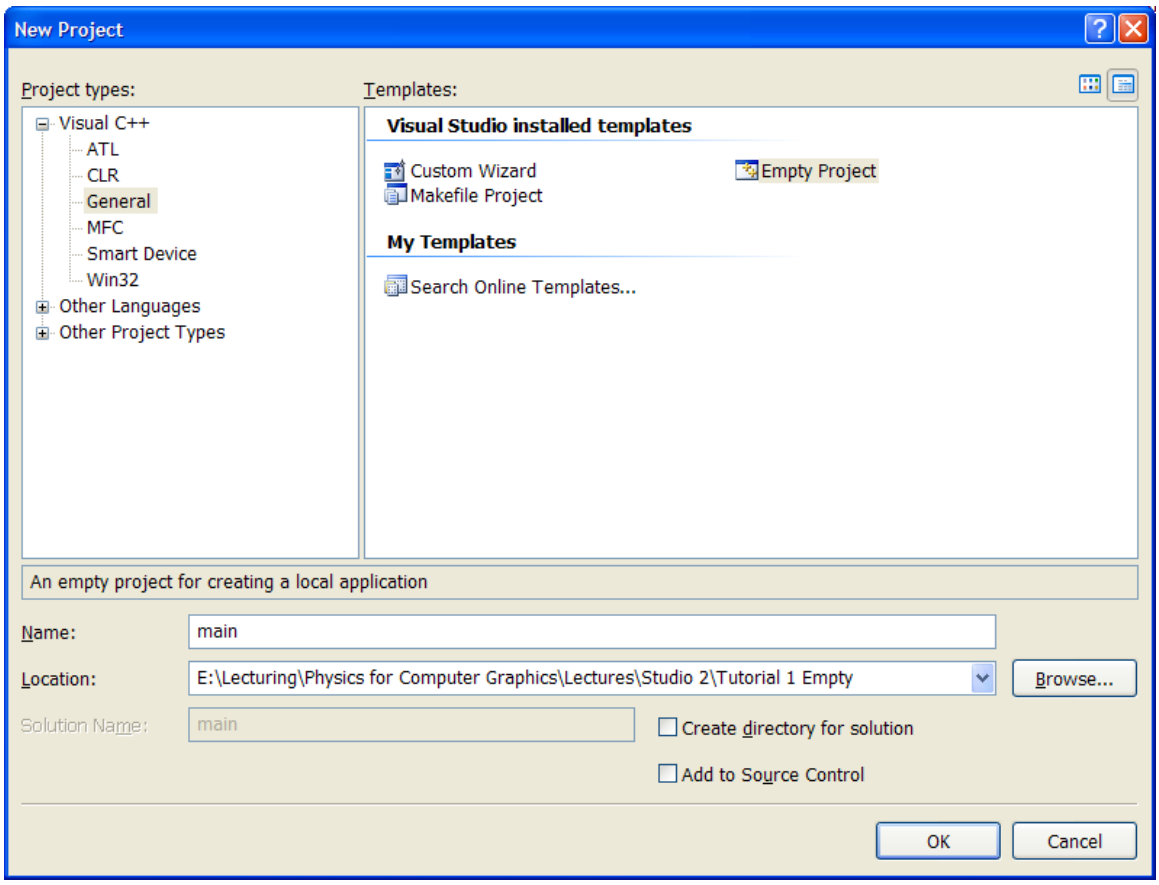
Next, follow the wizard and leave the default options as shown from the screenshots below.


Win32 Application Wizard - test
Welcome to the Win32 Application Wizard
Overview
Application Settings
These are the current project settings:
Console application
Click Finish from any window to accept the current settings.
After you create the project, see the project's readme.txt file for information about the project features and files that are generated.
< Previous
Next >
Finish
Cancel


Win32 Application Wizard - test
Application Settings
Overview
Application Settings
Application type:
Windows application
Console application
DLL
Static library
Additional options:
Empty project
Export symbols
Precompiled header
Add common header files for:
ATL
MFC
< Previous
Next >
Finish
Cancel

Congratulations, up not now you have created your first C++ program without writing any code. To see what you have done, press *Ctrl+F5* to execute the application. When you do that you should get an empty MS-DOS window.

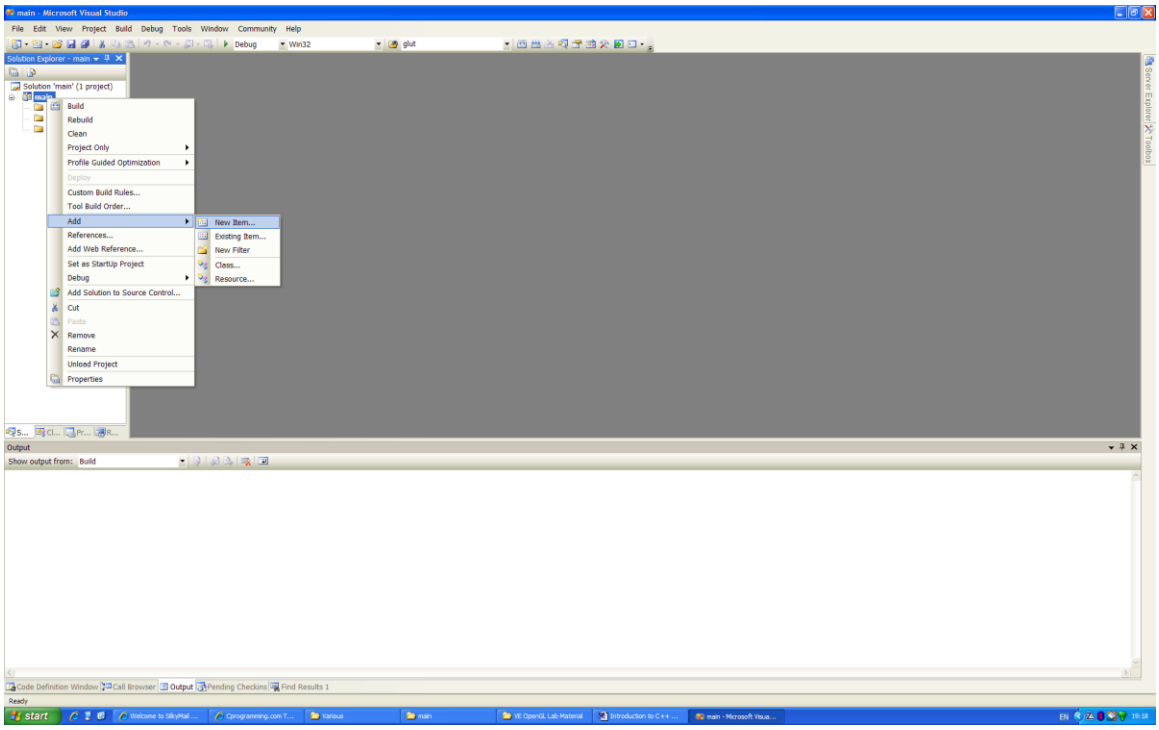
However, if you prefer to write everything from scratch you can do it! Close that workspace by selecting *File->Close Solution*. Try creating a new project (*File->New Project*) but in this case select General->Empty project, as shown from the screenshot below.



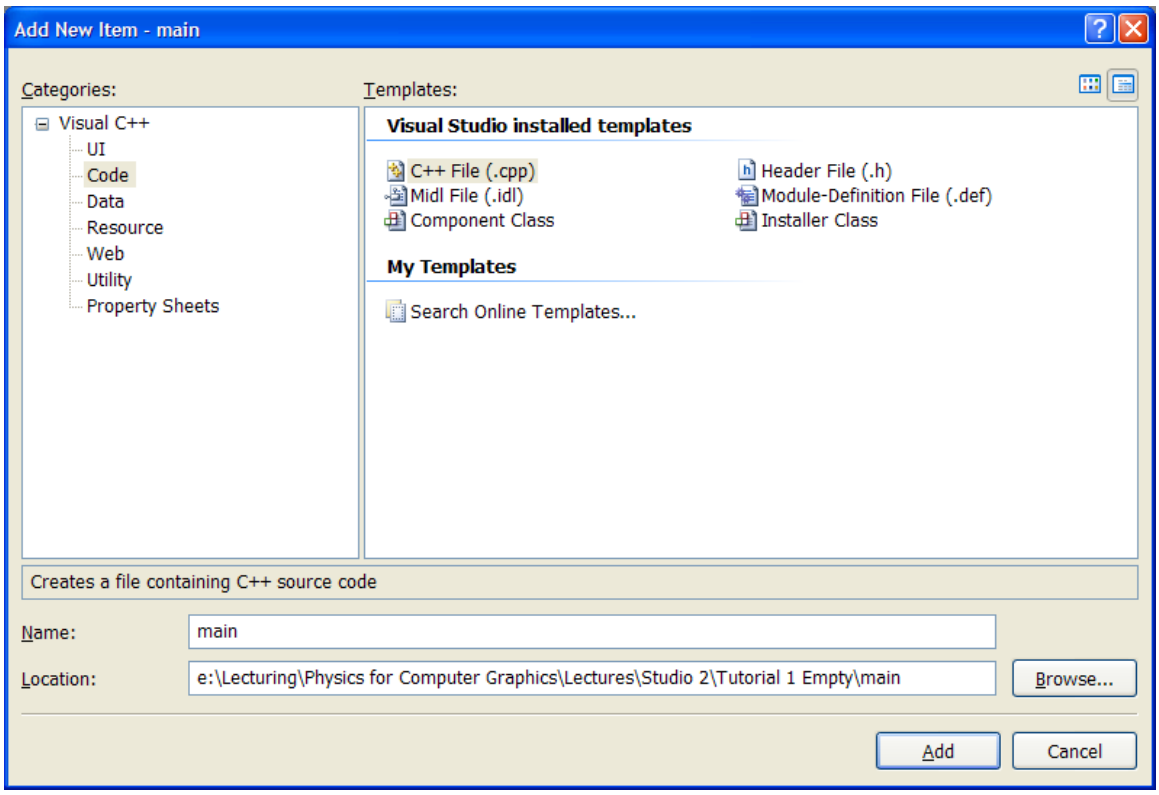
Input a new directory name (call it Tutorial 1 Empty) and specify a name (call it main). Right-click on the *main* solution (which can be found on the Solution Explorer) and select the *Rebuilt Solution* from the drop-down menu. You should get a message like the one shown below:

```
1>------ Rebuild All started: Project: main, Configuration: Debug Win32 ------
1>Deleting intermediate and output files for project 'main', configuration 'Debug|Win32'
1>main - up-to-date
========== Rebuild All: 1 succeeded, 0 failed, 0 skipped ==========
```

Next right-click again on the main solution and select the *Add->New Item* option as shown from below.



Then from the menu, select *C++ File (.cpp)*, and choose the name *main* again as shown below.



If you try to 'Rebuilt' the solution, you will get an error since there is nothing included in the *main.cpp* file. The next step is to add the necessary code to open the MS-DOS window again. This can be done by typing:

```
int main()
{
}
```

Try to rebuilt and execute the application by pressing *Ctrl+F5*.

Now it is time to start writing some code!

Example 1: Print a String on the Screen

First include the standard C++ library (iostream) and add some comments. To include the iostream library type the following:

```
#include <iostream>
using namespace std;
```

Note that we use cin.get() which is another function call: it reads in input and expects the user to hit the return key. So, inside the main function type the following code:

```
cout<<"PHYSICS FOR COMPUTER GRAPHICS! My first C++ program!\n";
cin.get();
return 1;
```

Now, rebuilt and execute the solution. Change the text string to something else. You can also try the same code in the Win32 console application.

Example 2: Get User Input

In this example we need to demonstrate how to get input from the user and print it on the screen. One of the first things that we need to do is to define a variable, which will hold the data. In this example, we call the variable *number* and declare it as integer type. The function cin>> reads a value into number; the user must press enter before the number is read by the program. cin.ignore() is another function that reads and discards a character. Remember that when you type input into a program, it takes the enter key too. We don't need this, so we throw it away.

```
int number;

cout<<"Please enter an interger number: ";
cin>> number;
cin.ignore();
cout<<"User has entered: "<< number <<"\n";
cin.get();
```

```
return 1;
```

Now, rebuilt and execute the solution.

Example 3: If Statements

Without a conditional statement such as the *if* statement, programs would run almost the exact same way every time. If statements allow the flow of the program to be changed, and so they allow algorithms and more interesting code. Moreover, a *true* statement is one that evaluates to a nonzero number. A *false* statement evaluates to zero. When you perform comparison with the relational operators, the operator will return 1 if the comparison is true, or 0 if the comparison is false. To get a better idea type the following code.

```
int age;

cout<<"Please input your age: "; // Enter the age
cin>> age;                       // The input is stored in age variable
cin.ignore();                 // Throw away enter
if ( age < 100 )                 // If the age is less than 100
{
      cout<<"You are pretty young!\n";
}
else if ( age == 100 )
{
      cout<<"You are old\n";
}
else
{
      cout<<"You are really old\n";// Executed if no other statement is
}
cin.get();

return 1;
```

Now, rebuilt and execute the solution.