EXPERIMENT 1
MICROSOFT VISUAL STUDIO AND C PROGRAMMING

Aims

1. Learning primary functions of Microsoft Visual Studio
2. Introduction to C Programming
3. Running C programs using Microsoft Visual Studio

In this experiment we will be using Microsoft Visual Studio 2008; however, with small differences, usage of most integrated development environments are similar.

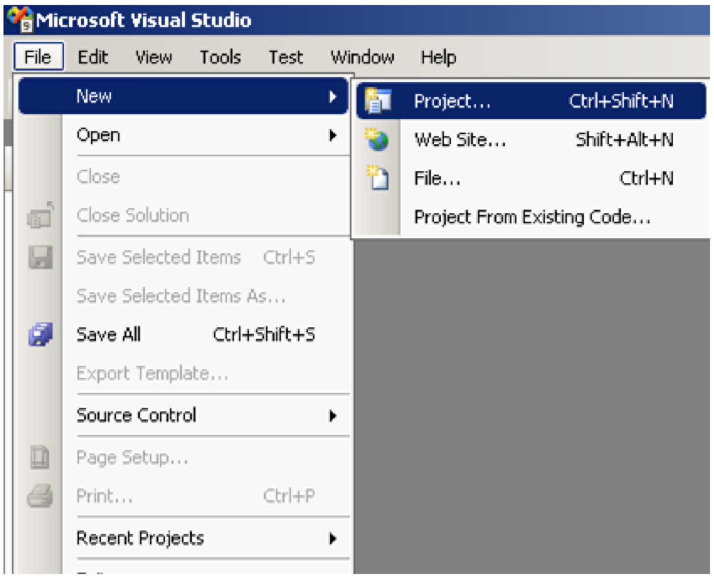
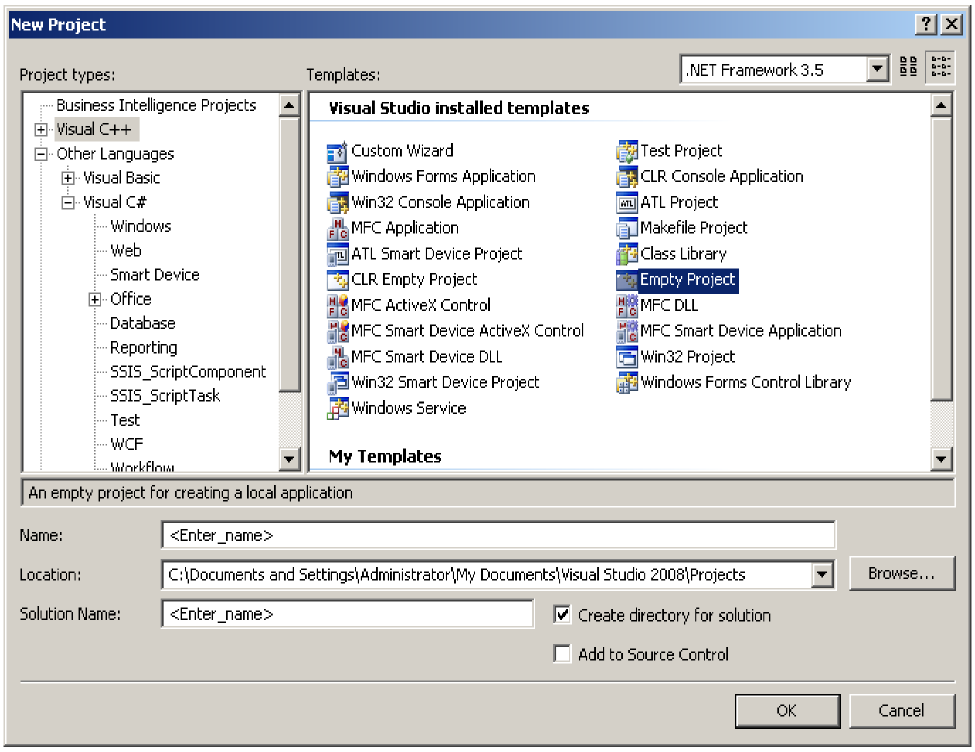


Figure 1: File menu

Creating a New Project

To create a new project, start Visual Studio 2008 (Start → Programs → Microsoft Visual Studio 2008 → Microsoft Visual Studio 2008). Select Project from the New list contained in the File menu, as shown in Figure 1. This operation will display a dialog screen shown in Figure 2. Select "Visual C++" from the "Project types" list shown on the left. Select "Empty project" from the right list and give a name to your project. (Note that you may need your project later on; therefore, choose a suitable name that you can remember). These steps will create an empty project for you to work on.



An empty project for creating a local application

Name: <Enter_name>

Location: C:\Documents and Settings\Administrator\My Documents\Visual Studio 2008\Projects

Solution Name: <Enter_name>

☑ Create directory for solution

☐ Add to Source Control

Figure 2: New project window

The first step to do in a new project is to create a file that we can write our codes into. Firstly, find "Project explorer" window. If it is hidden, you can open it from "View" menu. Right click on empty part of Project explorer and choose "New item" from "Add" menu as shown in Figure 3. This operation will open the window shown in Figure 4. From this window, choose "C++ file" and supply its name to create your file.

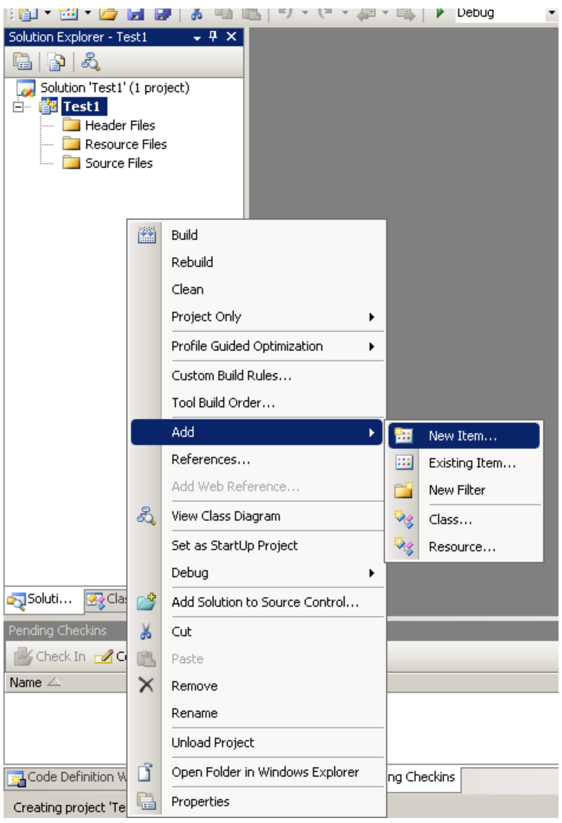
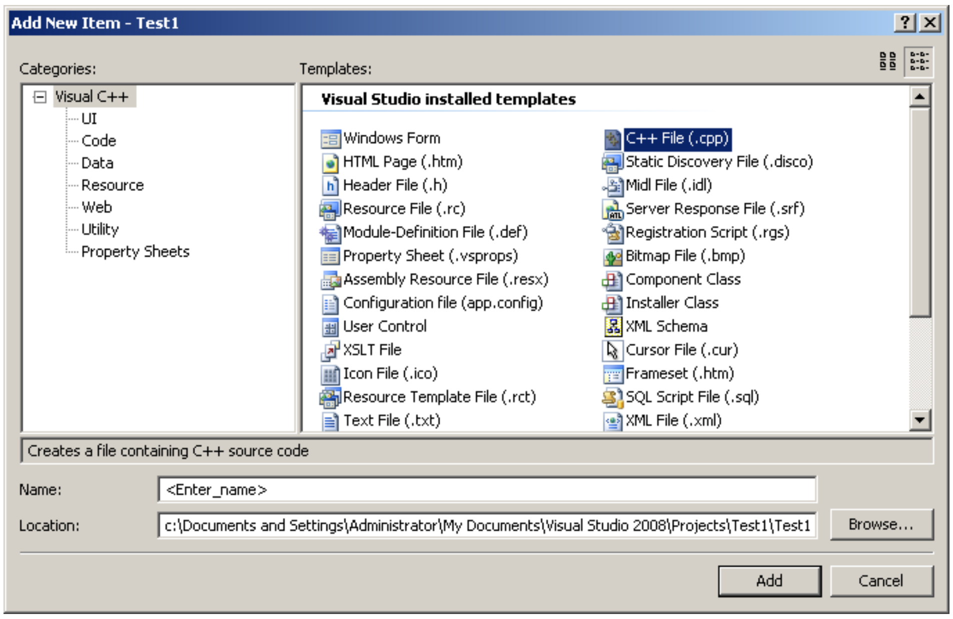


Figure 3: Add file menu


Add New Item - Test1

Categories:

Visual C++
- UI
- Code
- Data
- Resource
- Web
- Utility
- Property Sheets

Templates:

Visual Studio installed templates

Windows Form	C++ File (.cpp)
HTML Page (.htm)	Static Discovery File (.disco)
Header File (.h)	Midl File (.idl)
Resource File (.rc)	Server Response File (.srf)
Module-Definition File (.def)	Registration Script (.rgs)
Property Sheet (.vsprops)	Bitmap File (.bmp)
Assembly Resource File (.resx)	Component Class
Configuration file (app.config)	Installer Class
User Control	XML Schema
XSLT File	Cursor File (.cur)
Icon File (.ico)	Frameset (.htm)
Resource Template File (.rct)	SQL Script File (.sql)
Text File (.txt)	XML File (.xml)

Creates a file containing C++ source code

Name: <Enter_name>

Location: c:\Documents and Settings\Administrator\My Documents\Visual Studio 2008\Projects\Test1\Test1

Figure 4: New item dialog

First C Program

Note: Performing these operations as you read them will help you to understand it better.

The first part that should exist in a C program is its entry function. This function should be named as "main", so that C compiler will know which method it has to call. This function should return an integer (int) after its operation. For now we will only use the value 0, which denotes that the application completed its task as expected. The following is the shortest C program that can be executed.

```
int main()  {
    return 0;
}
```

As a simple program we can use "Hello World!" program which will print "Hello World!" to the screen and exits. To perform this operation we will require standard input/output library. We can perform include operation with the following preprocessor directive. This command should be written in global scope, which resides outside the function and other structure definitions. Moreover, the preprocessor directives, which starts with # symbol, does not need semicolon (";") afterward. Generally, preprocessor directives are place at the top of your file.

```
#include <stdio.h>
```

After adding this library file to our project, we can use **printf** (print formatted) function to print text to the screen. This function is defined in "stdio.h" file. The following code fragment will print "Hello World!" to console screen. This code should be inside our main function. However, since our program ends at **`return 0;`** line, printing code should be before that line. The “\n” added to the end of the text denotes that the text coming afterward will be printed on the next line.

```
printf("Hello World!\n");
```

In Visual Studio 2008, the console application that we write automatically closes after they are finished. Because of this, the result will not be available for us to read. We will use “pause” command from MS-DOS to prevent this. We can use “system” command to run programs from the operating system. However, we will need to include related library, which is standard library (stdlib.h). Therefore, following line should be added to our program. As before, it should be on top.

```
#include <stdlib.h>
```

The following command runs the pause program which will wait for user to press a key.

```
system("pause");
```

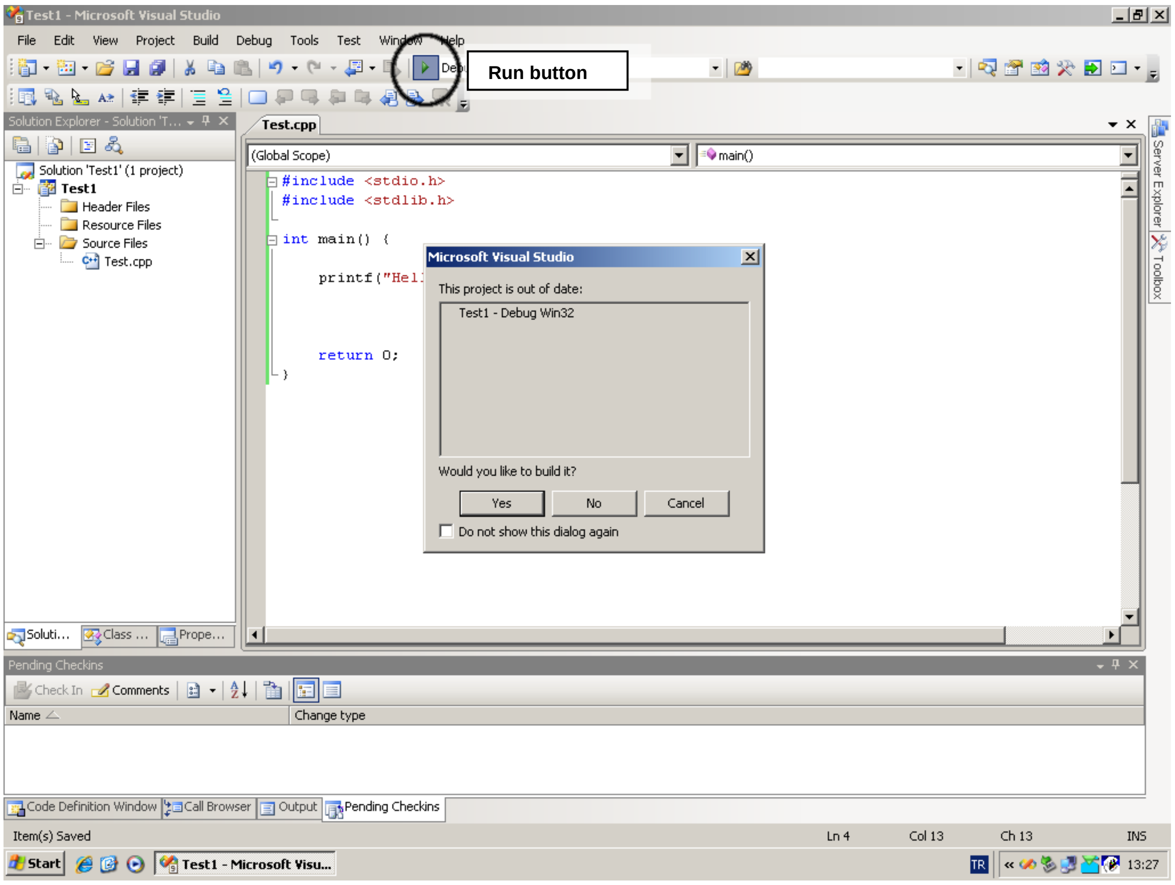
This program stops our program execution until a key is pressed. Therefore, it should be after the operation we perform. However, it should be before the end of our program, otherwise it will not be effective. You can use this command to pause your program temporarily on other occasions too.

The following is the whole program.

```
#include <stdio.h>
#include <stdlib.h>

int main() {
   printf("Hello World!\n");


   system("pause");

   return 0;
}
```

Running Your Program

To run your program you can press the button shown in Figure 5 or “F5” key. After issuing run command, the window shown in Figure 5 will be displayed. This dialog window asks if we want to compile our program. Since we want to run the codes we typed we have to compile our program. Therefore, the answer should be “Yes”. If there are no errors in your program, a console screen will be displayed. If your program requires input, you can enter it through this screen. Also your program output will be shown in here.



```
#include <stdio.h>
#include <stdlib.h>

int main() {

    printf("Hel

    return 0;
}
```

Microsoft Visual Studio

This project is out of date:

Test1 - Debug Win32

Would you like to build it?

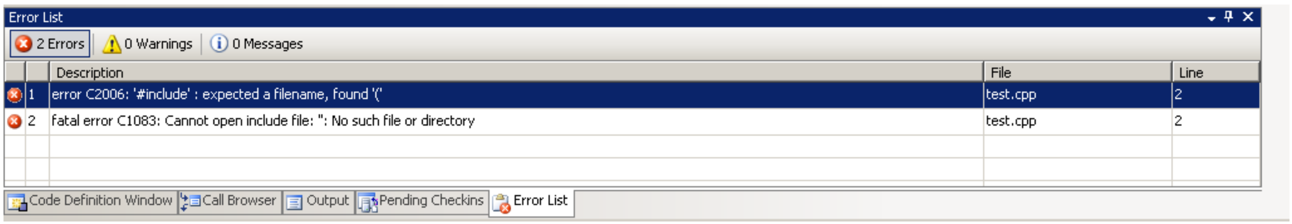
Yes | No | Cancel

Do not show this dialog again

Figure 5: Running your program

Fixing Problems

If there is an error in your program while compiling, Visual studio will ask if you want to run last working version of your program. Since we need to fix the problems we will answer “No” to this question and abort the running our program. After closing this dialog, a list of errors will be displayed. This window is shown in Figure 6. If you cannot see this window in your screen, you can open it using “View” menu. On the top of this list you will find “Errors”, “Warnings” and “Messages” buttons. These buttons control the visibility of these types of items. We are mostly interested in “Errors” and “Warnings”, so make sure that these two will be displayed. If you double click on an error or warning in this list, Visual Studio will take you to the line that is causing the error or warning. Some errors are caused by the general settings of your program; therefore, the line number information might not be available.



	Description	File	Line
1	error C2006: '#include' : expected a filename, found '('	test.cpp	2
2	fatal error C1083: Cannot open include file: '': No such file or directory	test.cpp	2

Figure 6: Error list

It is important to know that an error might be caused by previous line. If you are getting too many errors or if you are sure that the line displayed in the error list does not contain any problems, remember to check quotes, parentheses or semicolons (“;”) in the previous line. Most of the time missing semicolon error is given to the next line.

There are errors that cause your program to generate unexpected results. These are called logic errors and will not be handled by compiler. If you have such an error, examine your program carefully. These types of errors will be explained later.

The following is the list of most common errors. There might be errors that cannot be detected while your program is being compiled. Also common errors falling to this category is listed here.

1. Invalid symbol at include command

Error: Expected Filename...
After “include” preprocessor command, the file name should be written in angled brackets (< >).

2. Quotation errors

Error: unterminated string literal
In C programming language string literals should be surrounded by straight double quotes and each string should be written in a single line. Word processors replaces straight double quotes with opening and closing quotes, so be careful while copying from text documents.

3. main function is missing or defined multiple times

Error: undefined symbol __main or symbol __main is already defined
All C programs must have a main function. Check the name of it. In C language a function can be defined only once (there are exceptions to this rule) per project, therefore, you cannot add a second file and write another main function in there.

4. WinMain function is required

Error: undefined symbol __WinMain

You can write Windows applications in Visual Studio 2008. If you open such a project, the steps you should follow are different from what are shown in this lecture. Therefore, if you have this error, it means that you have chosen wrong project type. Create a new project and move your code to it.

5. Semicolon, coma or parenthesis missing

Error: Unexpected ... expecting ...

This error states that you have forgotten the specified symbol. Do not forget that this error is actually exist in somewhere before the error. Therefore, check the previous statements and even previous lines.