

# CSCI251 – Advanced Programming

## RUN FILES IN CODEBLOCKS

### Description

This document will instruct you on running a single file or multiple files on CodeBlocks. First, we introduce to you how a C++ program is edited and run. We then give you the instruction to run a single file. Finally, we guide you on how to make a C++ project, add the files to the project, and run the multiple files on CodeBlocks

### Learning Outcomes

By doing this hands-on instruction, you should be able to:

1. Understand the process of writing a C++ program
2. Create a new C++ file, edit, and run the code
3. Create a new C++ project, add files, and run the multiple files

**Run files in CodeBlocks****Table of Contents**

Description .....	1
Learning Outcomes.....	1
1. How a C++ Program Edited and Run.....	3
2. Run a Single C++ Program on CodeBlocks .....	5
Step 1: Create a new C++ File .....	5
Step 2: Edit Code and Run the Code .....	9
3. Create and Run Multiple C++ Files.....	13
Step 1: Create a New C++ Project and main.cpp .....	13
Step 2: Create a New C++ Header File (hi_Header.h) .....	14
Step 3: Create a New C++ Function File (hi_function.cpp) .....	15
Reference: .....	15

**Key Information:**

You will find exciting information and tutorials from this website: **<http://www.cplusplus.com/doc/tutorial/>**

## **1. How a C++ Program Edited and Run**

See figure 1. You will know how a C++ program is edited, compiled, and executed. Practically, we edit the code in the IDE like CodeBlocks, and then we click the build and run button to execute our code. However, behind this magic process, there is a sequence of 6 steps.

### **Step 1: *Code***

We start with writing the source codes (format .cpp). With a simple program, we write only one main file. However, in reality, we need to divide the codes into files. It will be stored in the library for reuse. In this case, we need to write other files like header files (.h).

### **Step 2: *Pre-process***

You always see the first line of the actual C++ code is `#include <iostream>`. It is in the preprocessing step. It always starts with a hash sign. Later, you want to give a name to a constant value, and you can use `#define`. These included files or symbol will be performed before running our actual code.

### **Step 3: *Compile***

In this step, the C++ compiler will compile all the code that we have written into the object codes (.obj, .o). We can check these files in the folder of the running files when we build the code.

### **Step 4: *Link Edit***

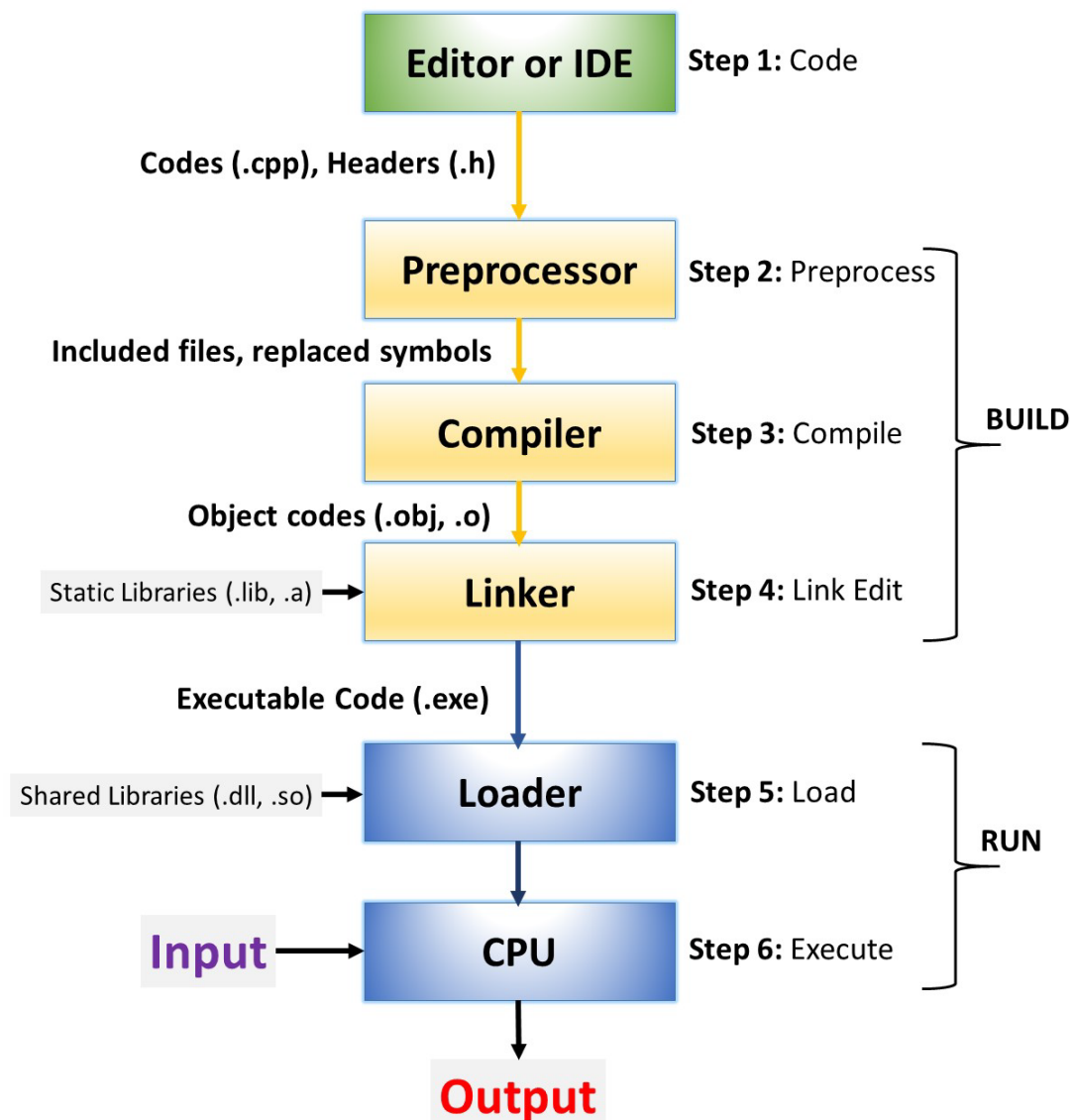
Other static libraries are performed to create object codes in this step. These objects will link with compiled object codes from step 3 to produce the executable code (.exe).

### **Step 5: *Load***

This step will load the executable code with shared libraries (.dll) into computer memory.

**Run files in CodeBlocks****Step 6: Execute**

It will run the executable code. In this step, the executable program will process the input to produce the output.



*Figure 1. Process of C++ Program Coded and Run*

From this process, we will recognize that we may have a program with one central file. However, we also have a project with multiple files.

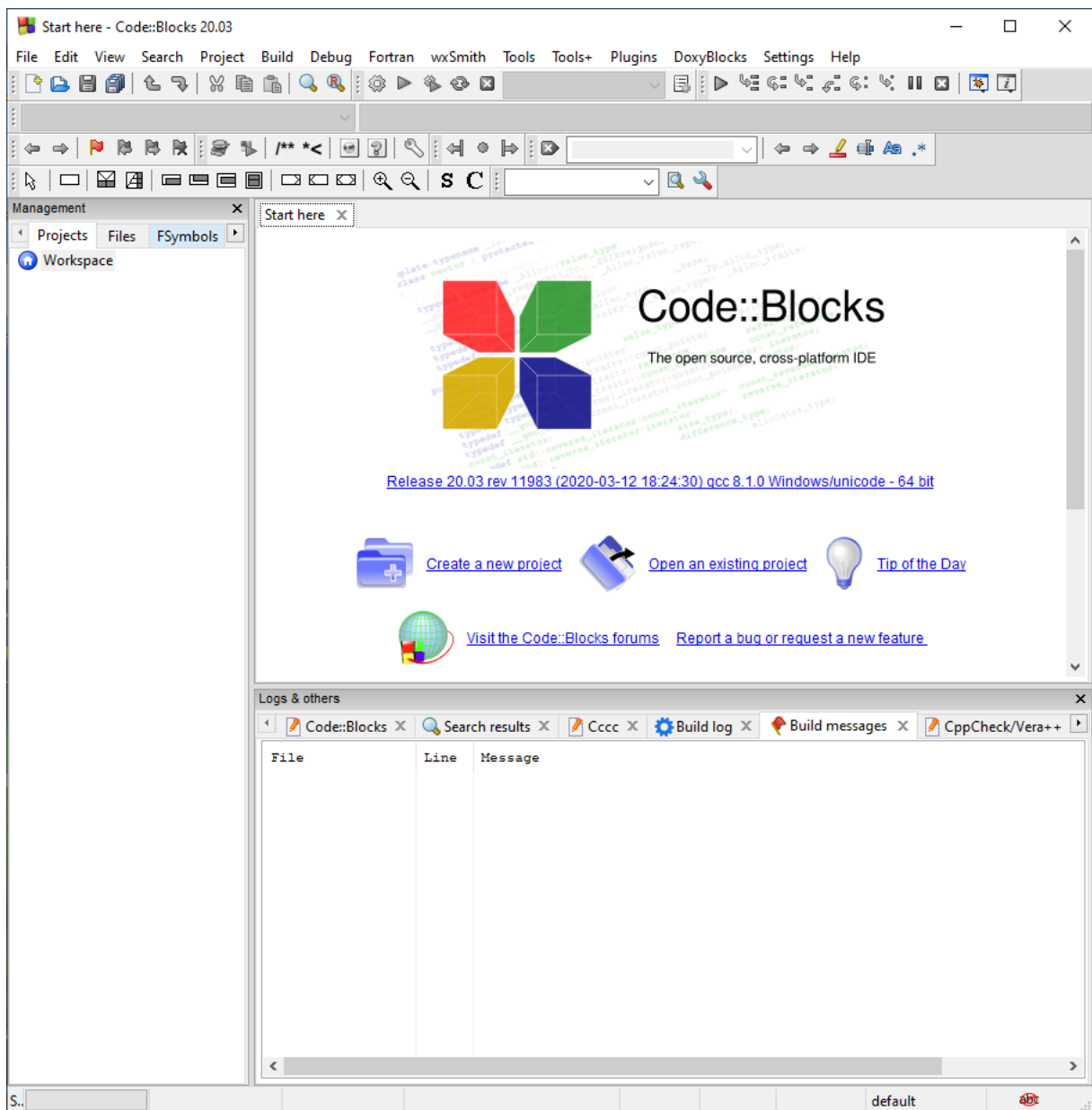
## Run files in CodeBlocks

### 2. Run a Single C++ Program on CodeBlocks

For this step, I use section 2 from the previous document about how to install CodeBlocks.

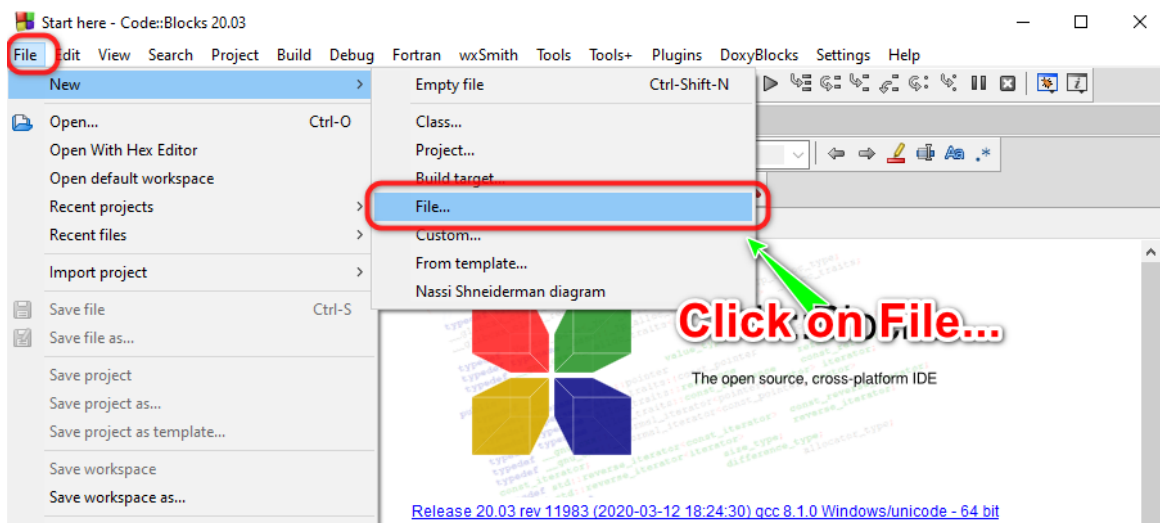
#### Step 1: Create a new C++ File

Please open CodeBlocks from the desktop or home menu. You can see the interface like the below picture. On the left, the management tab shows the tree of the file in the project or files. On the right, it is the main tab for editing the code. CodeBlocks support to create the software project, which includes many separate files. However, for the first time working with CodeBlocks, we should make a C++ simple file by following the next steps.

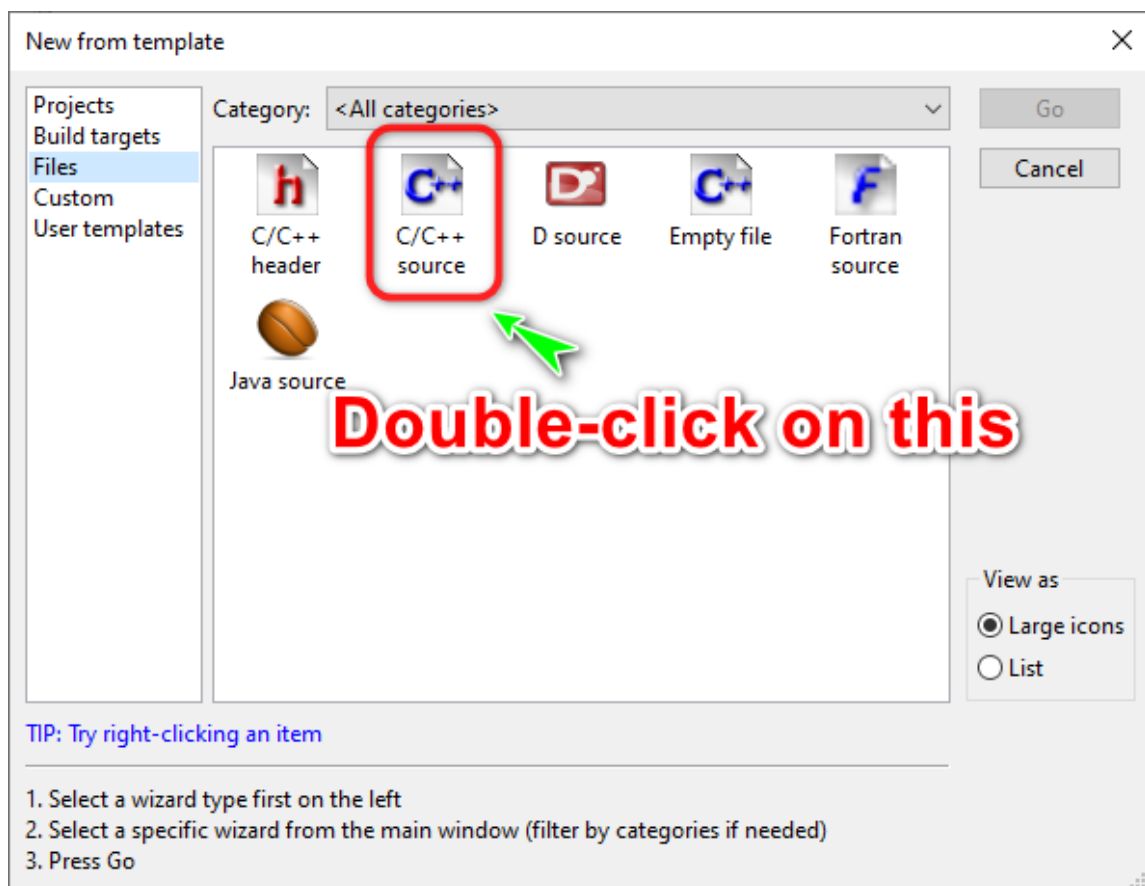


## Run files in CodeBlocks

Click on **File >> New >> File...**

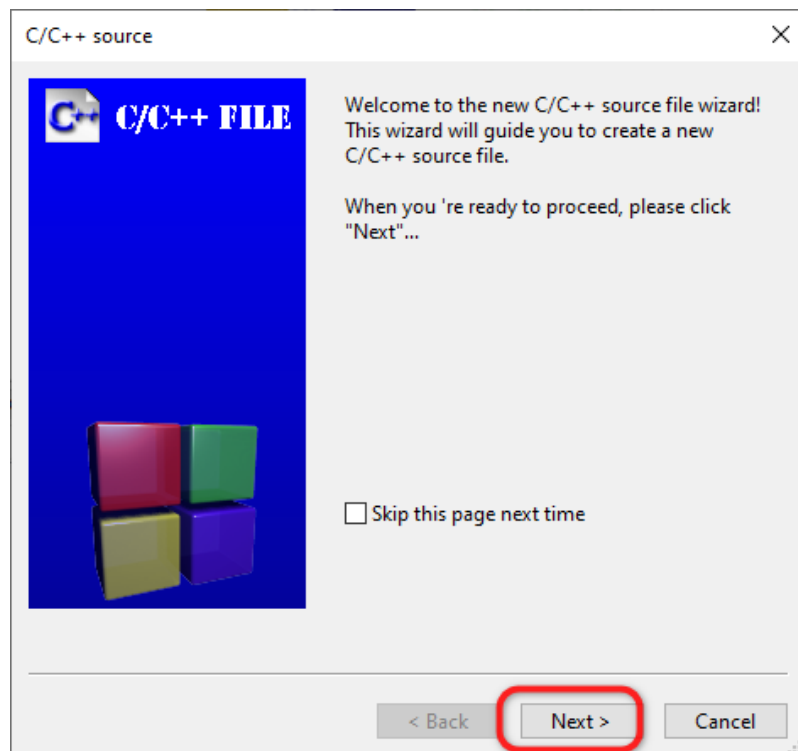


The “**New from template**” tab will pop up. Please double-click on the C/C++ source to create the source file.

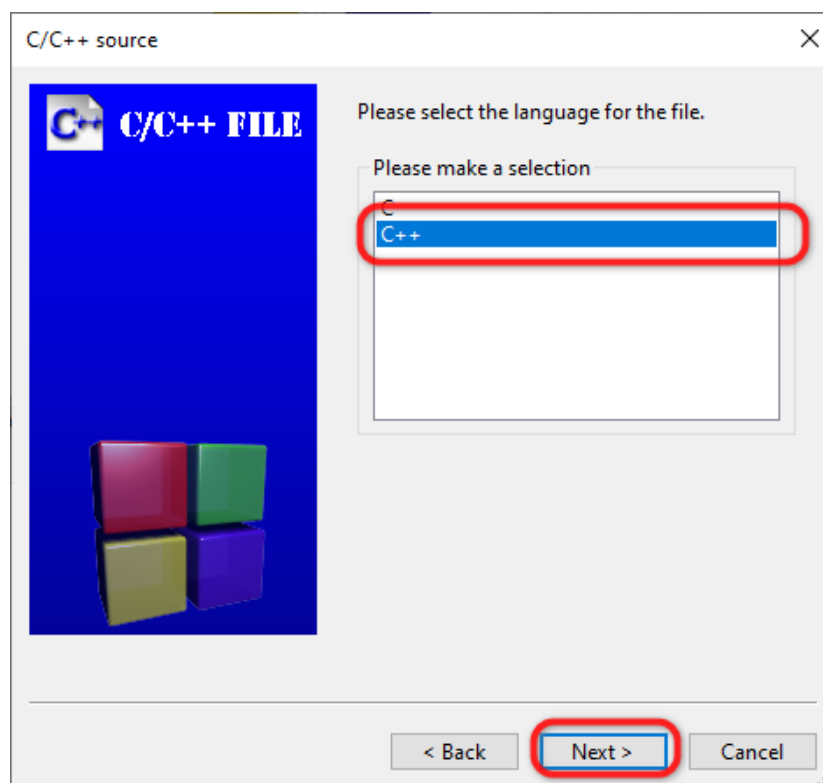


**Run files in CodeBlocks**

Then click “**Next**”

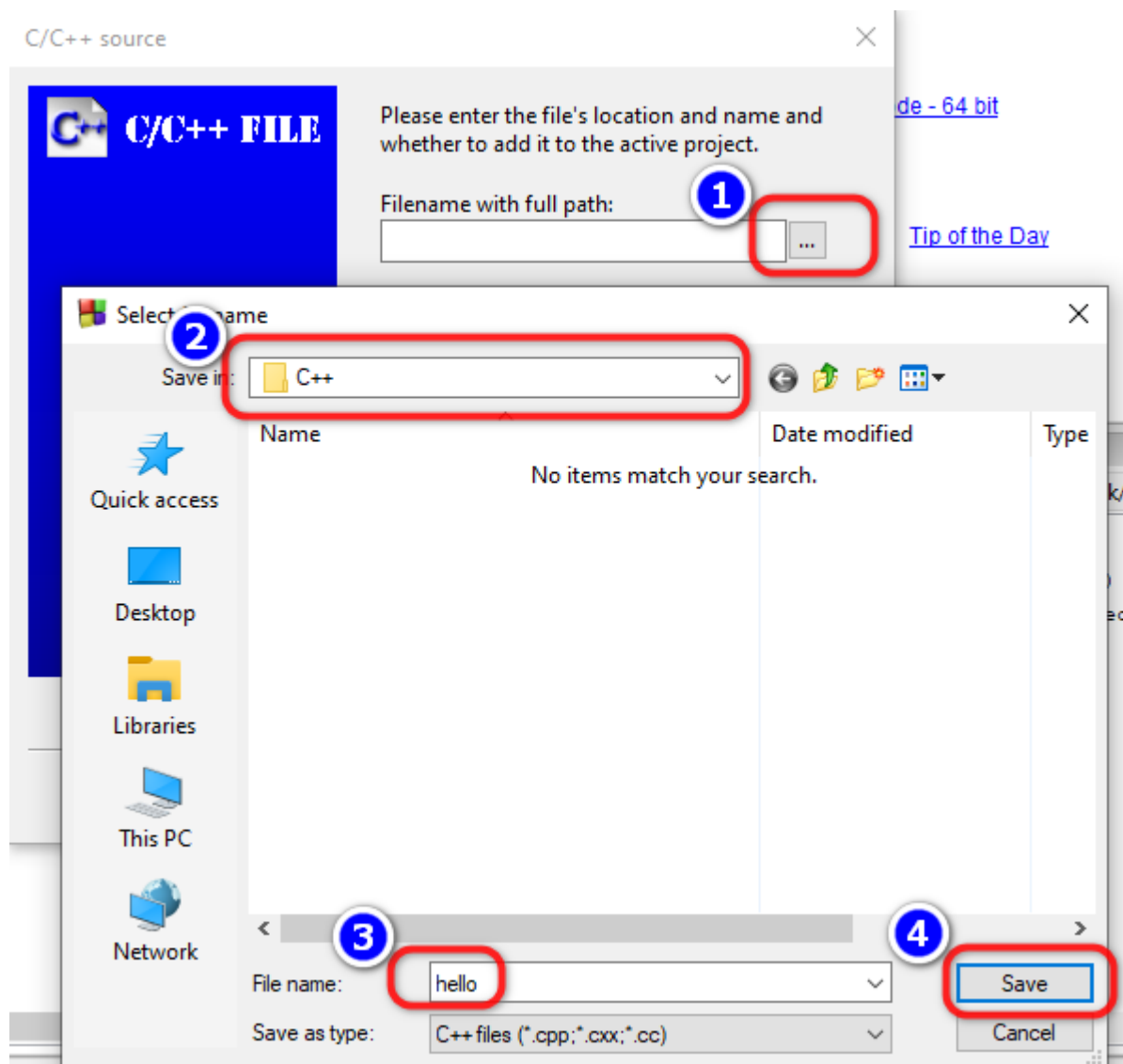


Choose **C++** from the text box, then click “**Next**”



**Run files in CodeBlocks**

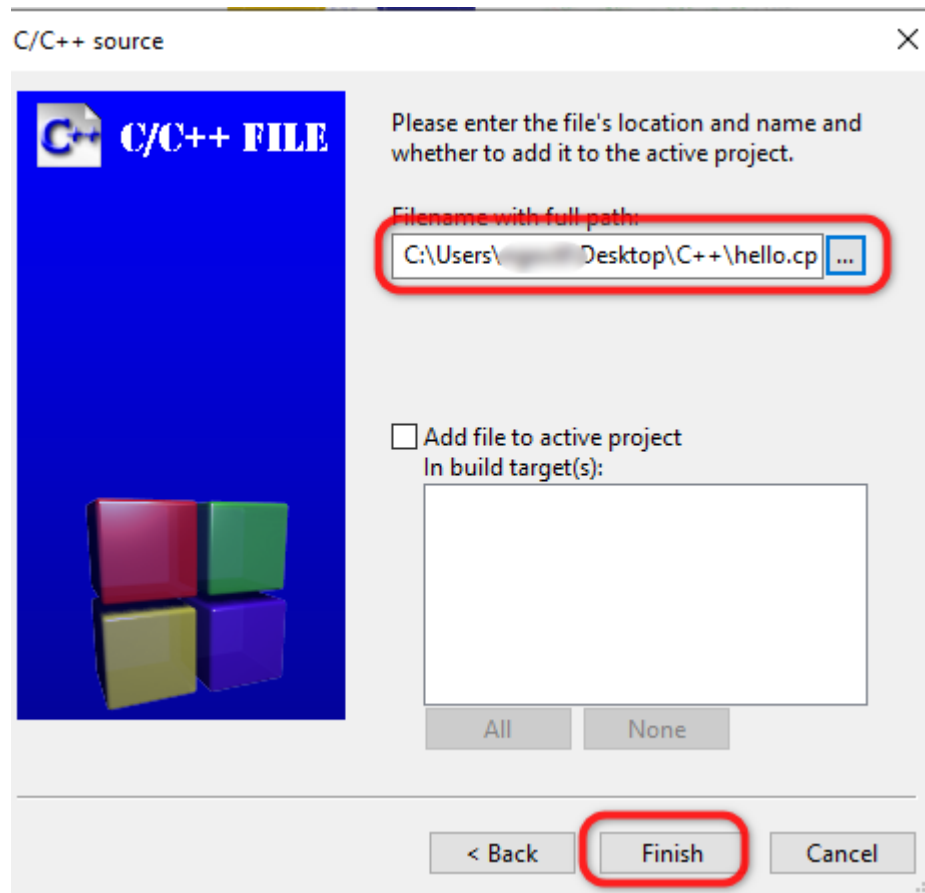
Browse to the folder you want to save the C++ file, then name for the file. In this example, the C++ source file has a name is “hello”, then click save





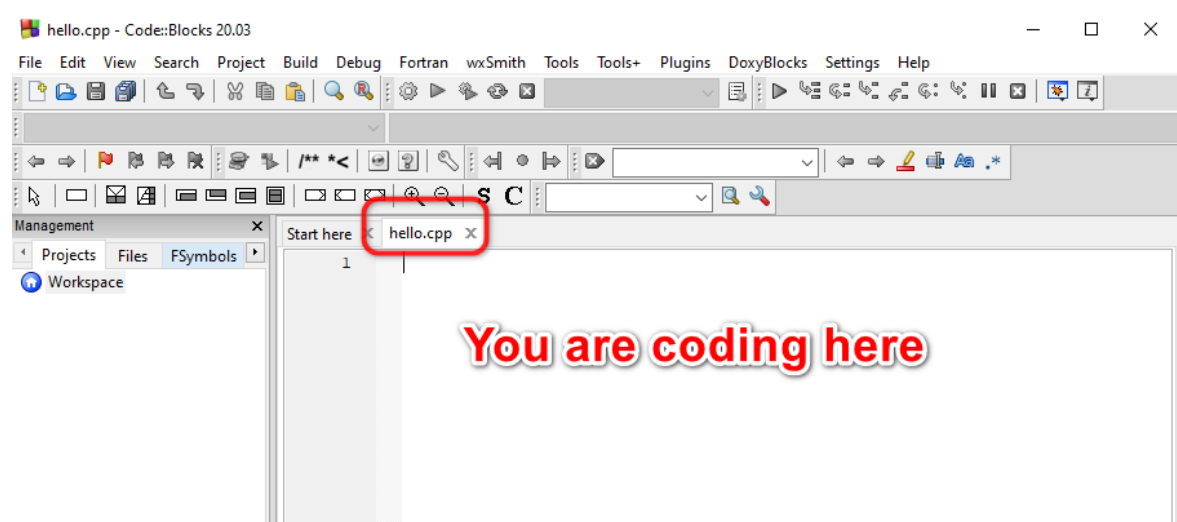
## Run files in CodeBlocks

Click **“Finish”** to create the hello.cpp file (cpp means C plus, plus).



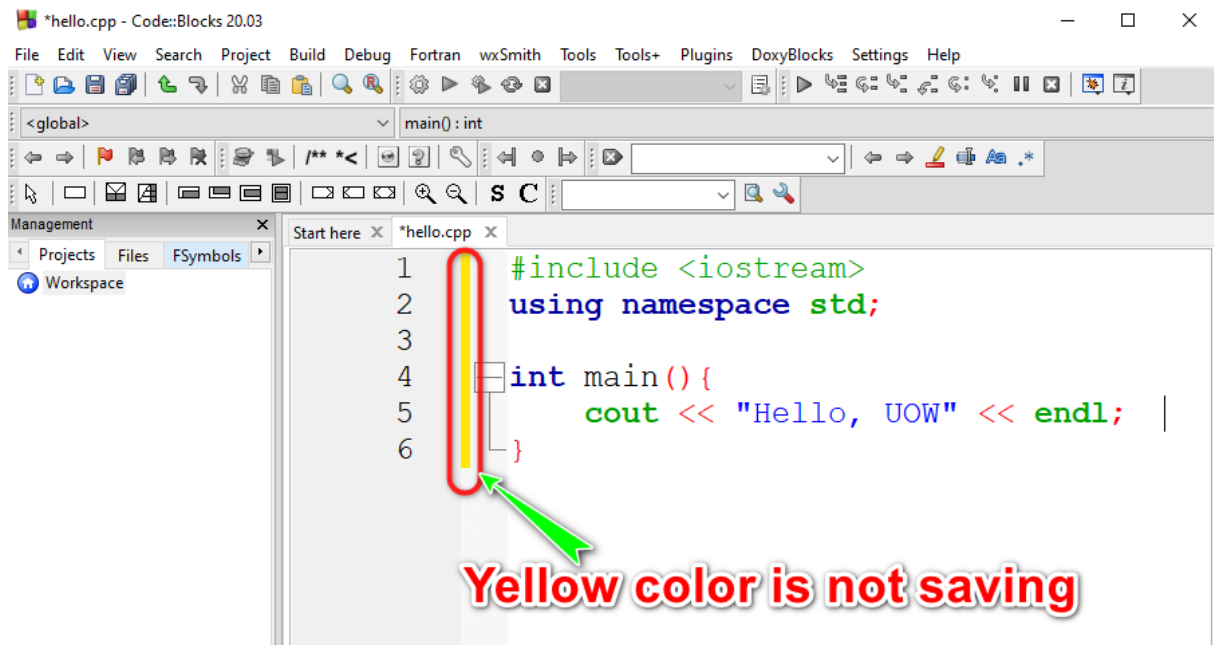
## Step 2: Edit Code and Run the Code

Then the main window will appear. Now you can edit your code in the main editing box.

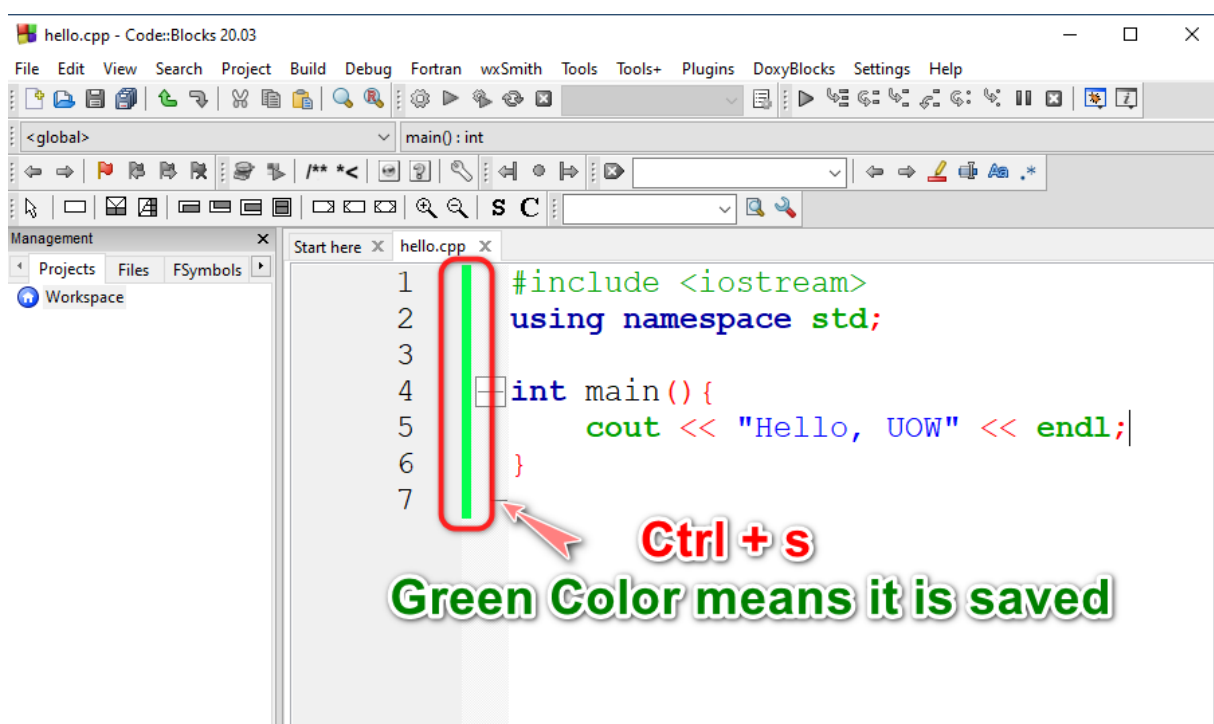


## Run files in CodeBlocks

Make a simple code like this. We want to show the text “Hello, UOW” to the screen. You can see the yellow stripe, which means the file is not saved.

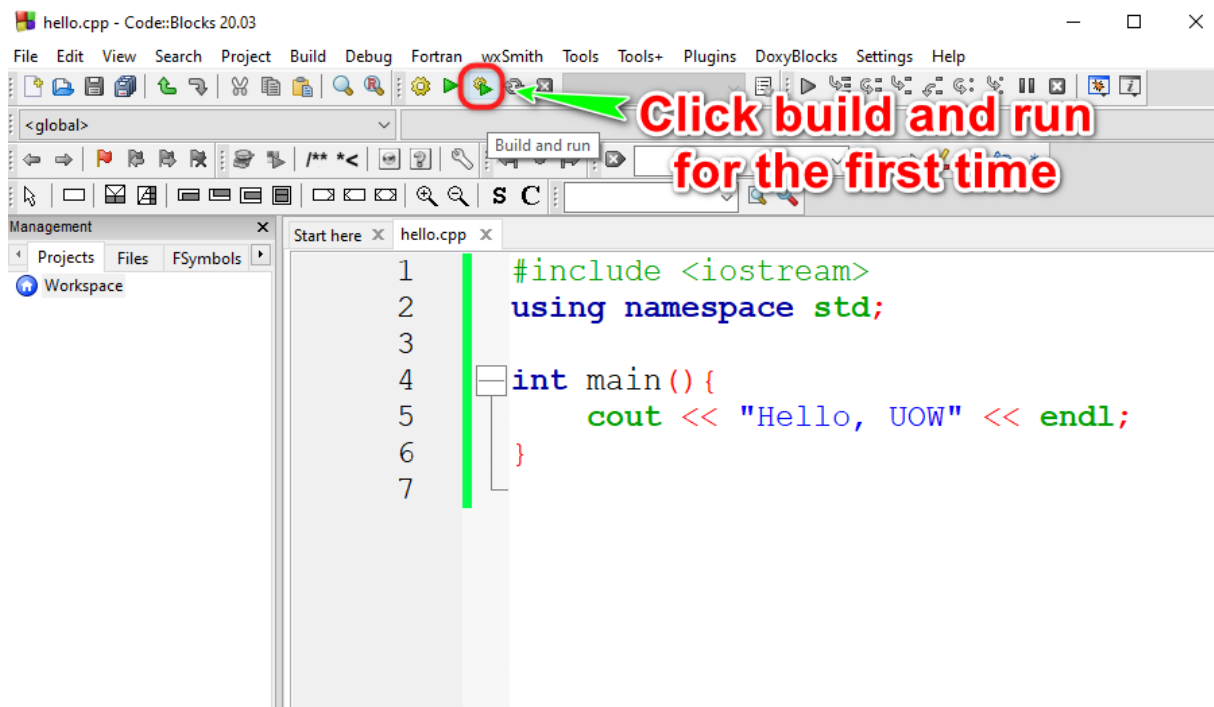


Press “Ctrl+S” to save the code, or you can click on the “save” symbol on the toolbar. The green strip means the file is saved for running.

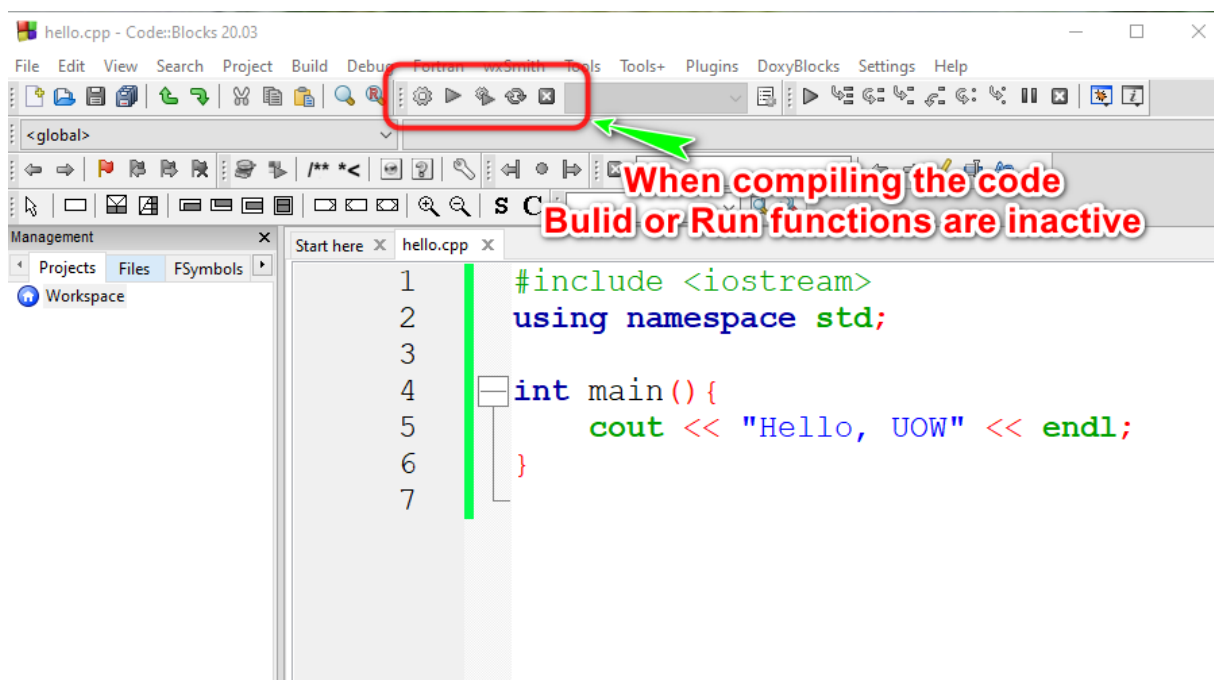


## Run files in CodeBlocks

To run the code, we have to build the code into the executable file then run this file to order the computer work. However, we can do these two steps by clicking the “Build and Run” button for the first time.

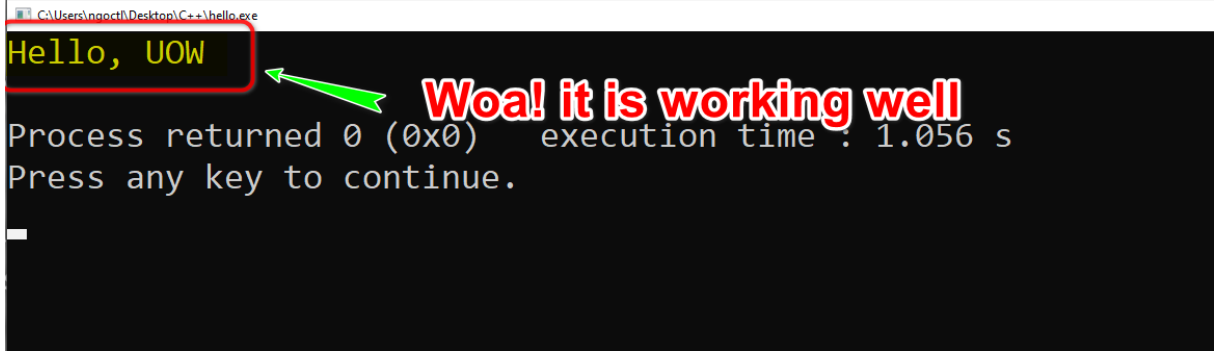


When compiling the code, build or run functions will be inactive.



**Run files in CodeBlocks**

The process of compiling is working and popping up the screen. It is successful if you can see the screen with the message “Hello, UOW”



```
C:\Users\naocti\Desktop\C++\hello.exe
Hello, UOW
Process returned 0 (0x0)   execution time : 1.056 s
Press any key to continue.

```

**Woa! it is working well**

### **3. Create and Run Multiple C++ Files**

As you know, now we rarely make software with only one file. Usually, to make the code clear and concise. We need to organize the codes into separate files. In simple C++, we have three popular files, including the main file (.cpp), function library file (.cpp), and a header file (.h). To manage all these files, we need to create a C++ project that contains all source files to run following the model in Figure 1. Remember that you will do all assignment by creating multiple files or creating a C++ project.

Now, you are quite familiar with the interface of CodeBlocks. Instead of a hands-on picture, I will instruct you by words. To make it easy to understand how multiple files run, I will do a project to run the same as an example from section 2. It means we create three source files with a simple task. It prints to the screen “Hello, UOW”. First, we will make the main file in the project. In this main file, we will include the header named hi\_Header (.h). In the main code, we will call the function hi\_UOW. Second, we will make the header file that includes the call to the function library name hi\_function (.cpp). Third, we will create hi\_function (.cpp) containing one function, `hi_UOW( )`, that prints the text “Hello, UOW”.

#### **Step 1: Create a New C++ Project and main.cpp**

- File ⇒ New ⇒ Project... ⇒ **Console Application** ⇒ Go.
- The "Console Application" wizard appears, then click
  - o Next
  - o Select "C++" ⇒ Next
  - o In "Project Title", enter "Hi" as the name of the project. In "Folder to create project in", set to the directory. For example “c:\user\Desktop\C++\project. Then you can leave other feature as default, then click ⇒ Next ⇒ Finish. In the Hi folder, we will have `Hi .cbp`. It is the CodeBlocks project file.

**Run files in CodeBlocks**

- Under the "Management" pane (left screen), ⇒ Choose "Projects" tab ⇒ Expand the project node "Hi" ⇒ Expand "Source" node ⇒ Double-click "main.cpp", which is a template program to say "Hello, world!". However, we edit the main.cpp file like this.

```
#include "hi_Header.h"

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

**Step 2: Create a New C++ Header File (hi\_Header.h)**

- File ⇒ New File... ⇒ Select **C/C++ header**
- C++ ⇒ Next
- In "Filename with full path" ⇒ Click the "Navigate" (...) button to navigate to the project directory (**Hi folder**) and enter the new file name (**hi\_header**). Check both the "Debug" and "Release" boxes (or "All") ⇒ Finish. By doing this, we have added a header file to the “Hi” project
- Under the "Management" pane (left screen) ⇒ Choose "Projects" tab ⇒ Expand the project node "Hi" ⇒ Expand "Headers" node ⇒ Double-click "**hi\_header.h**". It will create some default lines of codes. We add this line of code “void hi\_UOW( );”

```
#ifndef HI_HEADER_H_INCLUDED
#define HI_HEADER_H_INCLUDED

void hi_UOW();

#endif // HI_HEADER_H_INCLUDED
```

### Step 3: Create a New C++ Function File (hi\_function.cpp)

Do similar as we have created a header file. However, in this case, we generate a cpp file.

- File ⇒ New File... ⇒ Select **C/C++ source**
- C++ ⇒ Next
- In "Filename with full path" ⇒ Click the "Navigate" (...) button to navigate to the project directory (**Hi folder**) and enter the new file name (**hi\_function**). Check both the "Debug" and "Release" boxes (or "All") ⇒ Finish. By doing this, we have added `hi_function.cpp` file to the "Hi" project
- Under the "Management" pane (left screen) ⇒ Choose "Projects" tab ⇒ Expand the project node "Hi" ⇒ Expand "Source" node ⇒ Double-click "**hi\_function.cpp**". We add this code.

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

void hi_UOW(){
    cout << "Hello, UOW" << endl;
}
```

We save all the files and back to the main file, check all the code, then build and run the project. We will have the same result as in section 2. I know you will find it harder. However, all the big C++ project will be created and run this way. It is to create a C++ project.

### Reference:

<https://www.codeblocks.org>