

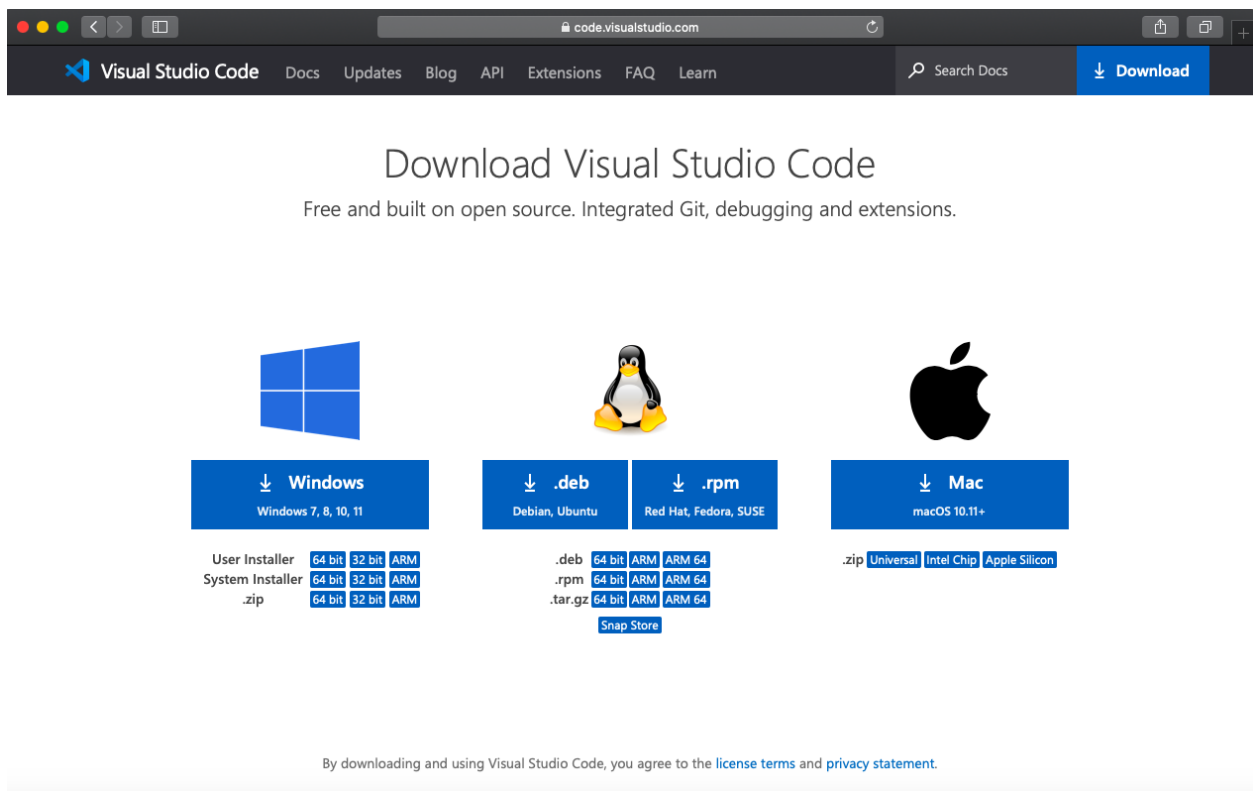
CSCI 1300 CS1: Starting Computing
Naidu/Yeh - Spring 2022
Visual Studio Code - MacOS

MacOS Installation Guide

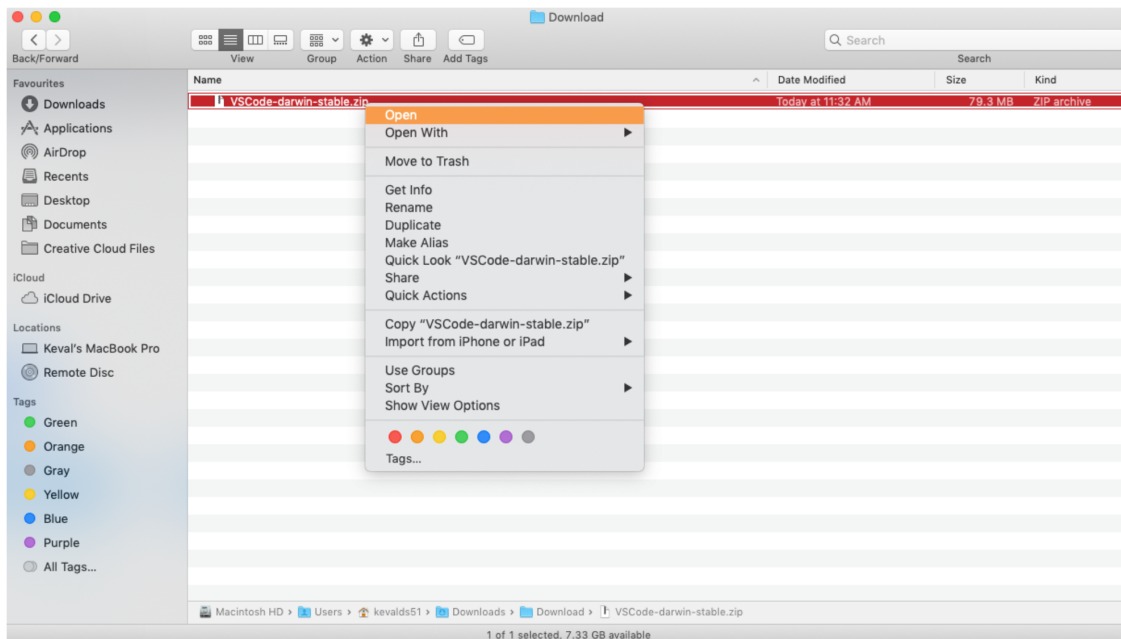
You will use Visual Studio Code (VS Code) to write and execute your programs locally.

Part 1- Install VS Code

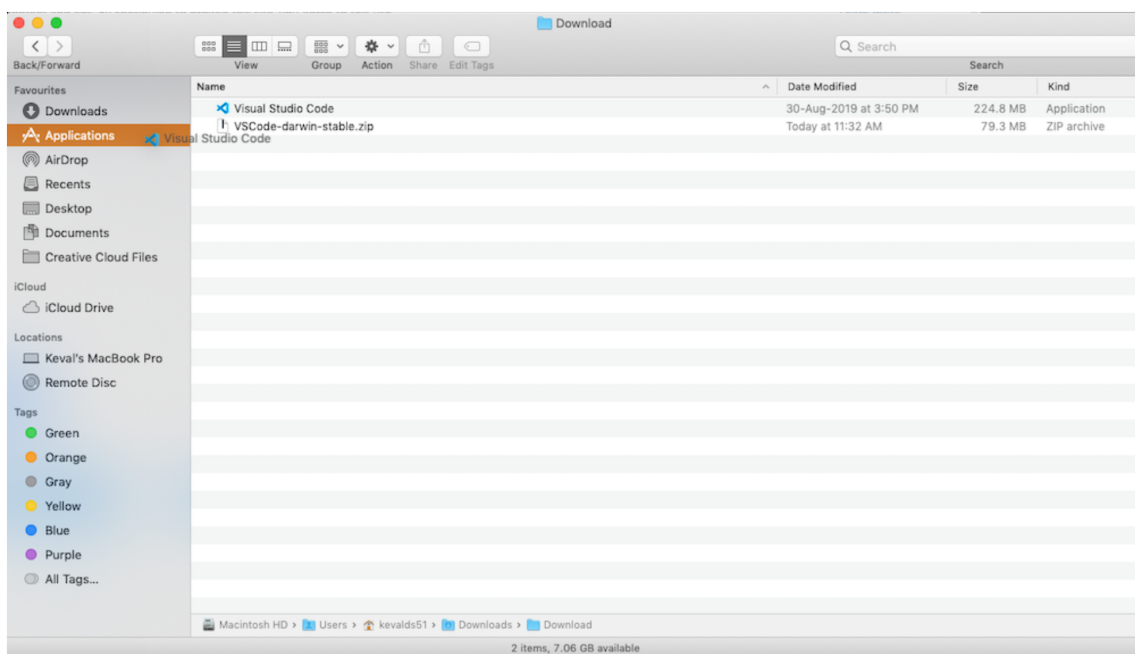
Step 1: Go to VS code [download page](https://code.visualstudio.com), and download for Mac.



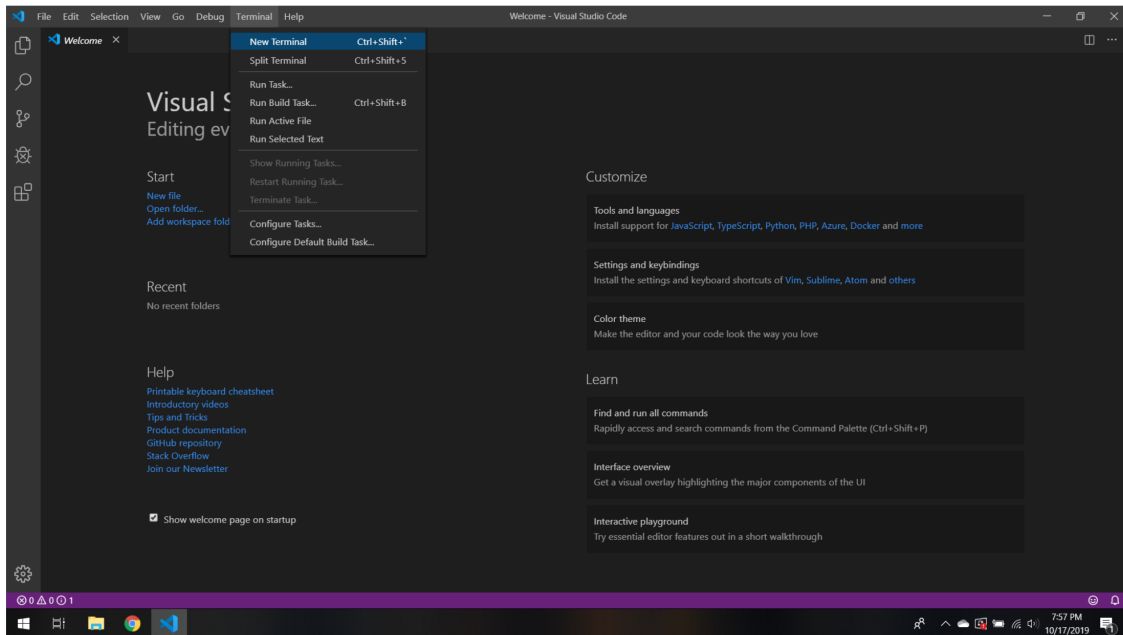
Step 2: After the download has finished, unzip the folder by double-clicking on it.



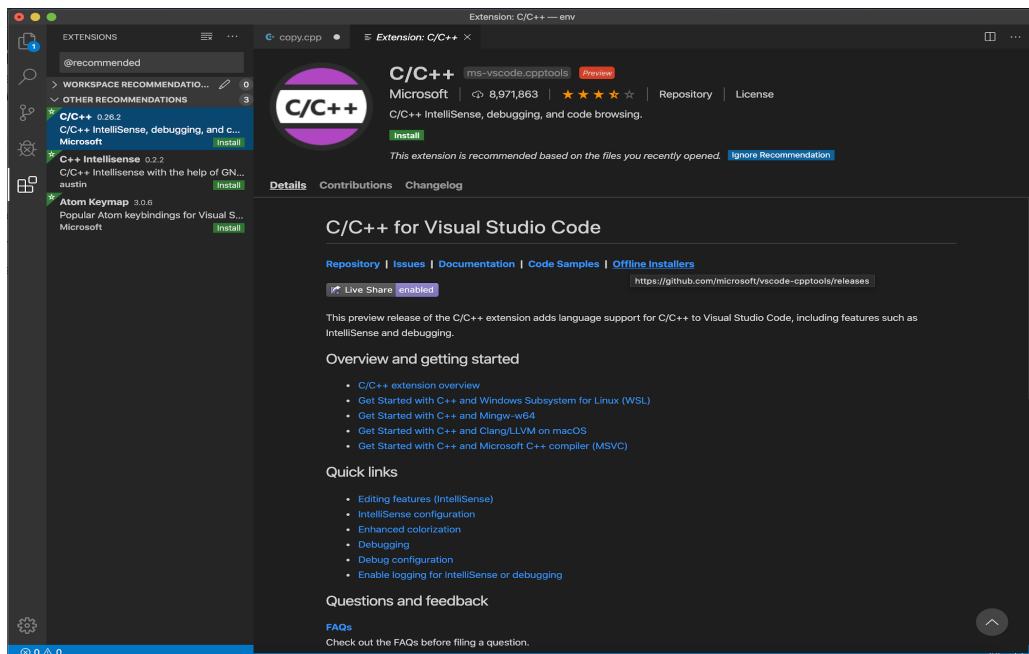
Step 3: Now you can see the “Visual Studio Code” application. Drag and drop this icon to the “Applications” folder of your computer.



Step 4: Double click on the "Visual Studio Code" icon to launch the application. (You might need to right click and select "open" if you cannot launch the program). Next, select the new "Terminal option" to open the terminal window.

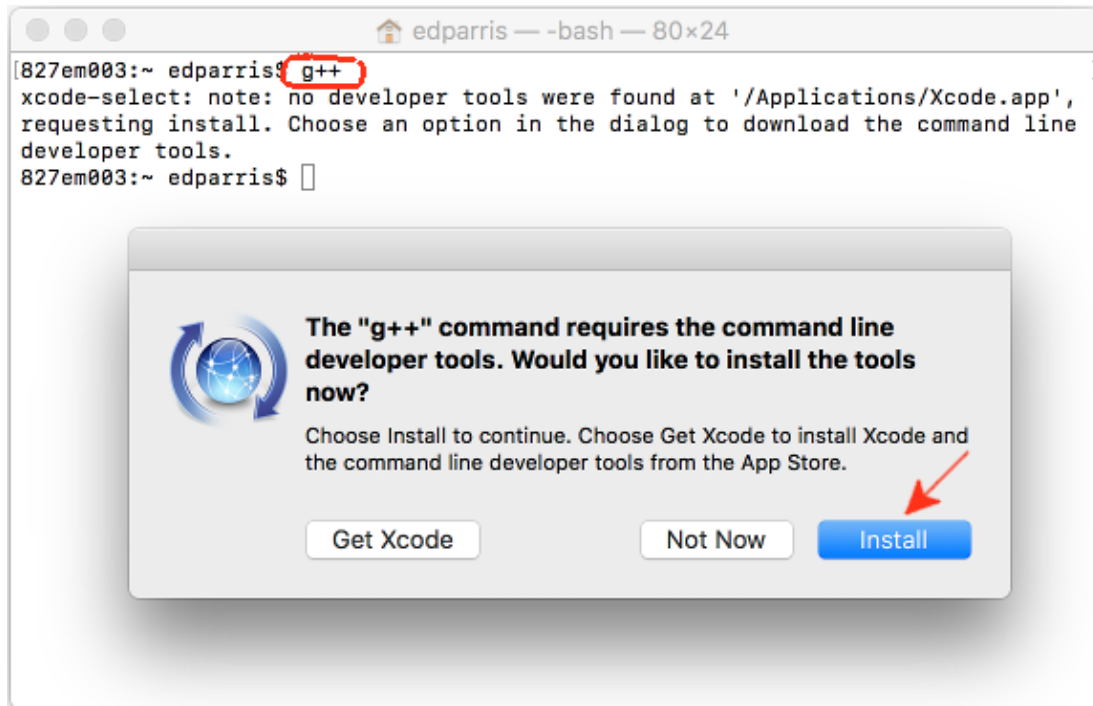


Step 5: Install C/C++ extension. In the toolbar on the left hand side of the screen click on the bottom icon for Extensions. Search for C/C++ and click install.



Part 2 - Install g++

1. Open a Terminal window.
(Press ⌘ Command+Space, type **Terminal** in the search field, and press the Return key.)
2. In the Terminal window type **g++** and press the Return key. We will see an alert box like this:



3. Choose **Install** to get only the command line tools unless you want to learn Xcode. Xcode can be installed later from the App Store.

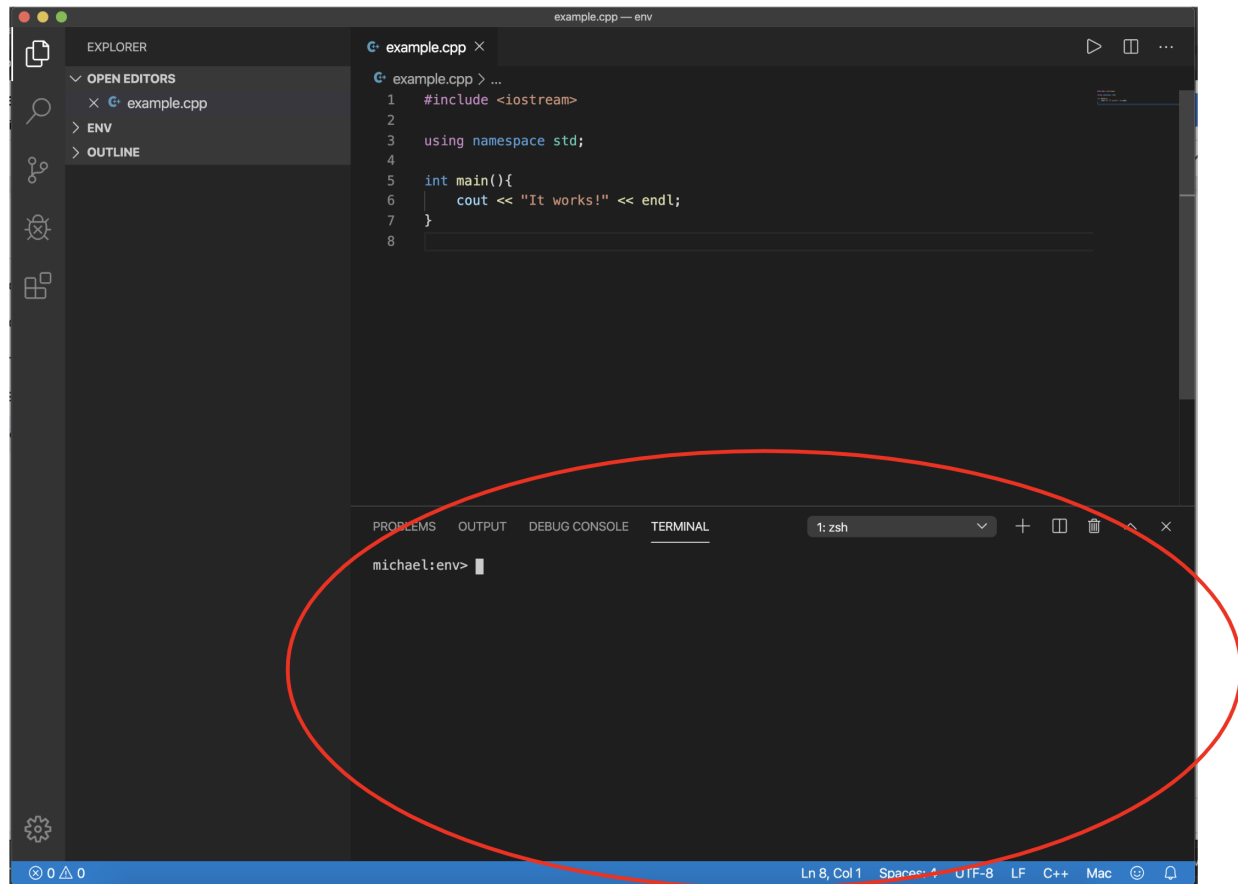
After installation, type **g++** in the Terminal, press the Return key, and verify the terminal prints the message, "no input files".

```
$ g++  
clang: error: no input files
```

Well done! Go to the section [Compiling Code In Terminal](#) to compile code on your computer.

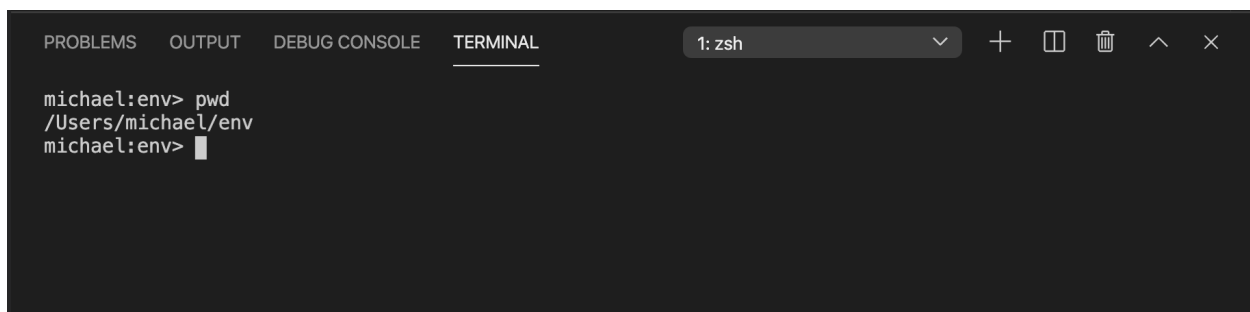
Compiling Code In Terminal

When you open VSCode at the bottom of the screen you should see a terminal.



In order to compile and run your code you first need to navigate to where your code is saved through the terminal. You do this using the **cd** (change directory) command (see the Homework 0 document on Canvas if you need an explanation on how to do this). If you have a folder open in VSCode by default the terminal will open to the location of that folder.

For instance on my computer I have a folder I made called **env** in VSCode which the terminal automatically opens to. You can check the current path your terminal is working at using the **pwd** command (print working directory) as shown below:

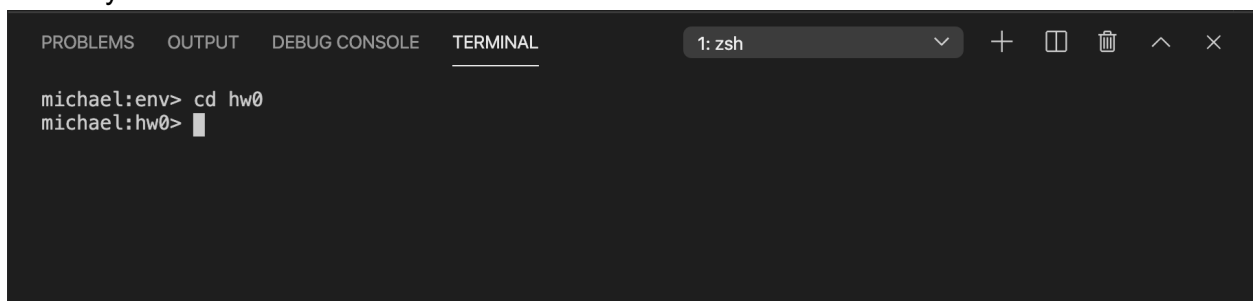


Note: Your terminal may look slightly different, as everyone's computer is different.

The first line in the screenshot above reads "**michael:env> pwd**", **michael** is the username of the person using the terminal, and **env** is the name of the current working directory. And **pwd** is the current command. Any command that users type will be after the **>** symbol. This symbol might be different depending on your computer, but that's okay.

Now the second line shows the output when the **pwd** command was run by pressing enter/return after the command which shows the current working directory. Don't worry about copying this directory path, yours can be different.

Now let's say the code I want to run is located in a subdirectory called **hw0** within the **env** directory, then I can navigate to **hw0** using the **cd** command (change directory). This command is of the form "**cd <destination>**". Where **<destination>** should be replaced by the path to the directory you want to switch to. Here if a folder/directory name is mentioned without a full path (as it is in the example below), it is considered a sub folder/directory of the current working directory.

A terminal window with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The terminal shows the command 'cd hw0' being executed, changing the directory from 'env' to 'hw0'.

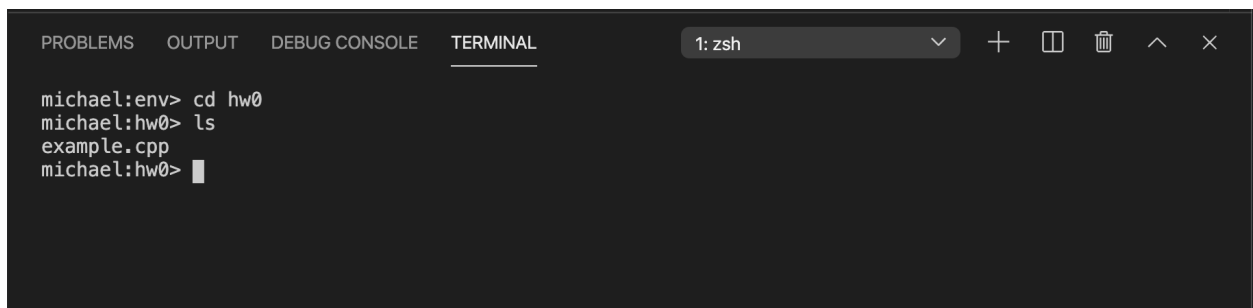
```
1: zsh
michael:env> cd hw0
michael:hw0> █
```

Instead, if you wanted to provide a full path, the command would look something like:

cd /Users/michael/env/hw0

which corresponds to the same directory in the above example, albeit using a full path to the directory.

Now let's double check the files we are looking for are actual there by using the **ls** (list) command.

A terminal window showing the 'cd hw0' command followed by 'ls', which lists the file 'example.cpp' in the current directory.

```
1: zsh
michael:env> cd hw0
michael:hw0> ls
example.cpp
michael:hw0> █
```

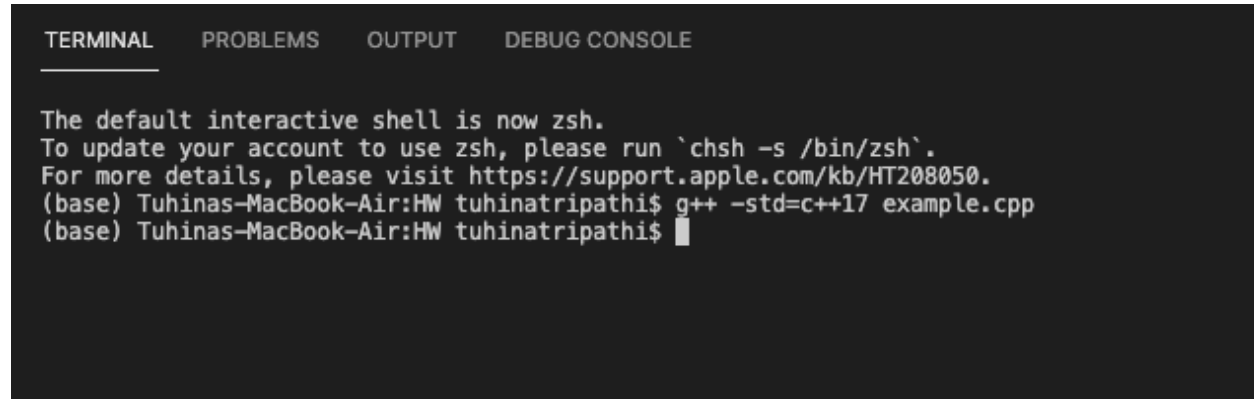
Now to compile our code we use the program **g++** in the following command.

g++ -std=c++17 example.cpp

g++ is the compiler program

-std=c++17 specifies the version of C++ we want to use

example.cpp is the file we want to compile



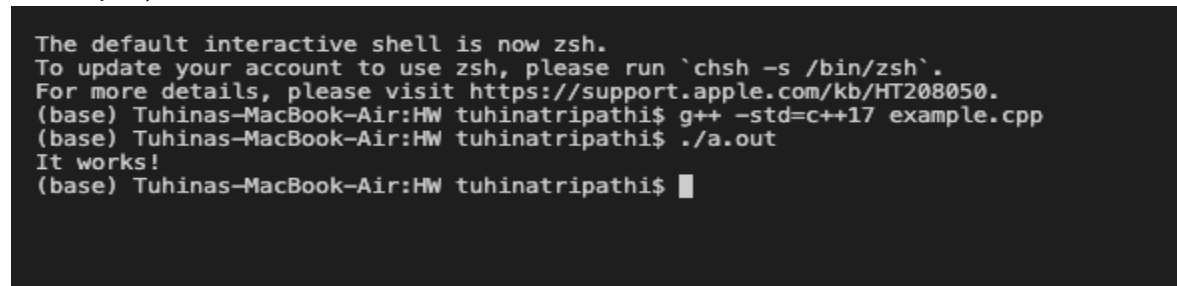
```
TERMINAL  PROBLEMS  OUTPUT  DEBUG CONSOLE

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) Tuhinas-MacBook-Air:HW tuhinatripathi$ g++ -std=c++17 example.cpp
(base) Tuhinas-MacBook-Air:HW tuhinatripathi$
```

This command creates a file named **a.out** which is the compiled version of the code in example.cpp. Now to execute our code we run a.out by typing the following and hitting enter.

./a.out

The same is shown in the screenshot below. (Note: the first line is from the previous command to compile)



```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) Tuhinas-MacBook-Air:HW tuhinatripathi$ g++ -std=c++17 example.cpp
(base) Tuhinas-MacBook-Air:HW tuhinatripathi$ ./a.out
It works!
(base) Tuhinas-MacBook-Air:HW tuhinatripathi$
```

And that is how you compile and run a C++ program from a terminal.