

CS 1436.0L1/003 – Local C++ Development Environment Setup (Windows)

1 Prerequisite

Before we are able to get your development environment set up, we first must make sure that your computer is able to run Windows PowerShell Scripts.

1.1 Windows PowerShell Setup

First, open Windows PowerShell by pressing your Windows key () and typing “Windows PowerShell”. Make sure that you open it as an administrator.

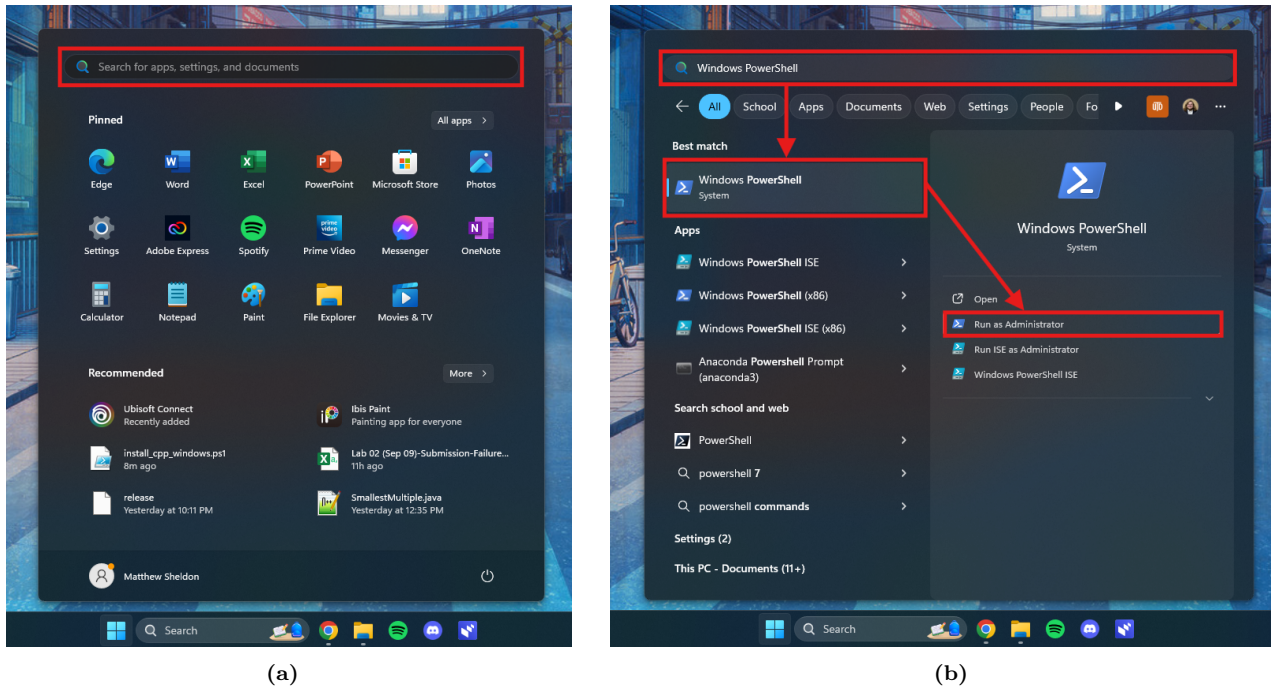





Figure 1. (a) Result of pressing , highlighting the text field for searching for apps.
(b) How to open Windows PowerShell as an Administrator.

Please note that you need to click on the “Run as administrator” button (the program will not run unless it has administrative privileges).

Then, type the following command into the opened Administrative PowerShell Terminal:

```
Set-ExecutionPolicy -Scope Process Unrestricted
```

When prompted to ensure that you want to change the execution policy, type the single character  and press . This will prevent you from having to answer  (“yes”) to potentially multiple prompts.

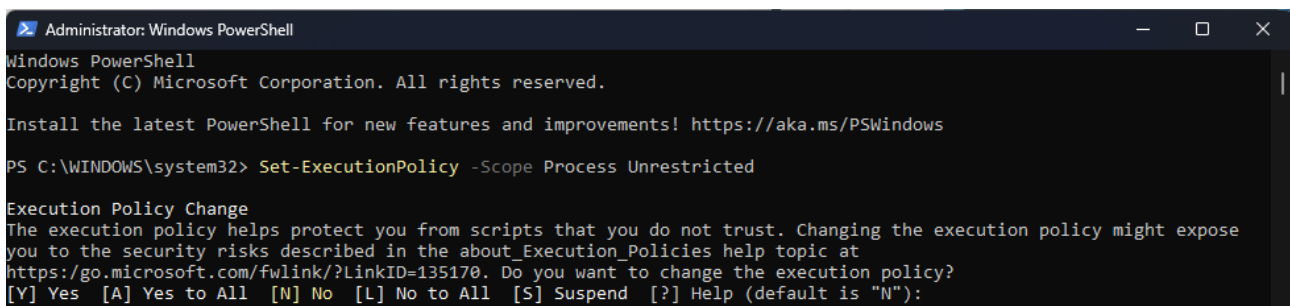


Figure 2. Prompt resulting from running the Set-ExecutionPolicy command

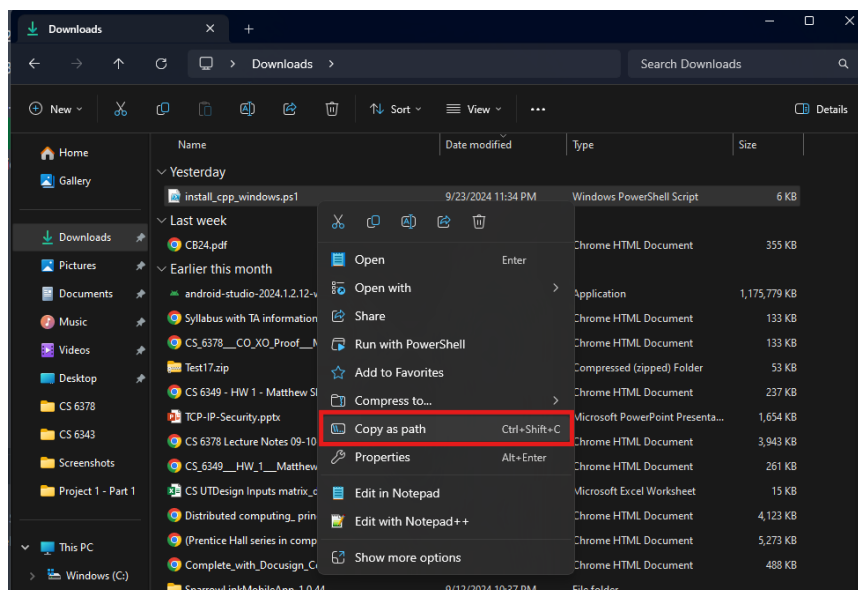
This will allow us to actually execute the script which will set up your C++ Development Environment automatically.

2 Setting up Your C++ Development Environment

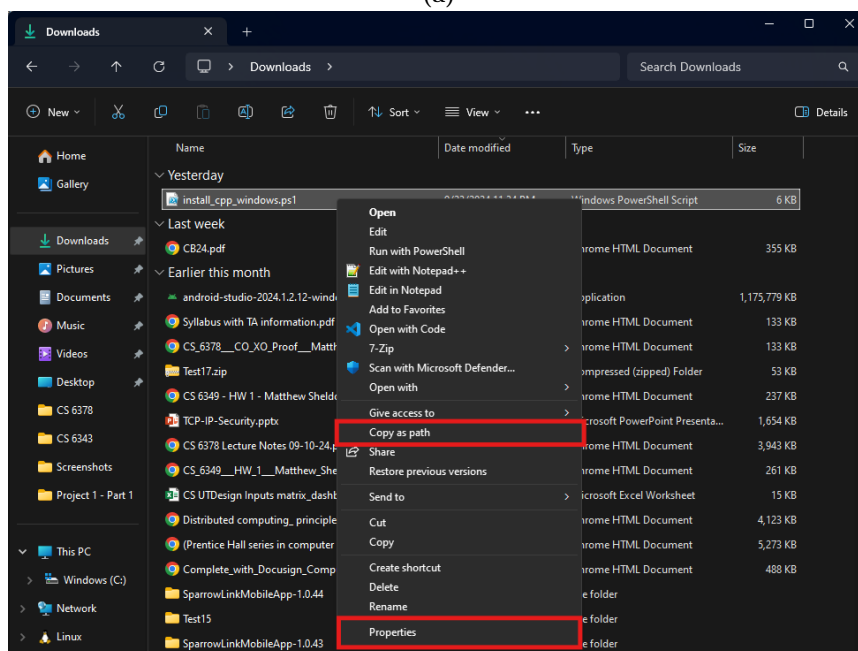
Now that we have enabled the use of PowerShell scripts, we will now set up your local C++ development environment using the provided script.

2.1 Locating the Script

When you downloaded the script provided to you, it likely ended up in your Downloads folder (unless you specified a different download location). Within Windows File Explorer, navigate to the directory/folder containing the script. From there, right click on the file, and select the option “Copy as Path”. If you are on an older version of Windows which does not have this option, then instead click on the “Properties” option. If you needed to click on the “Properties” button, at this point, the file path should be listed in a section called “Location” or something equivalent to that.



(a)



(b)

Figure 3. (a) How to copy a file’s location using the “Copy as path” method.
(b) How to get open the “Properties” of a file.

2.2 Running the Script

Now that you have copied that path to the script, in either the same Administrative PowerShell terminal as before or in a new one, type in the following command :

```
cd "<COPIED_PATH>"
```

Replacing <COPIED_PATH> with the path to the file. **If the path copied includes the file name itself, be sure to remove that from the path.** In other words, we just want the path to where the file is being stored, not the file itself. After pressing enter, you should see that you have navigated to the directory/folder containing the script:

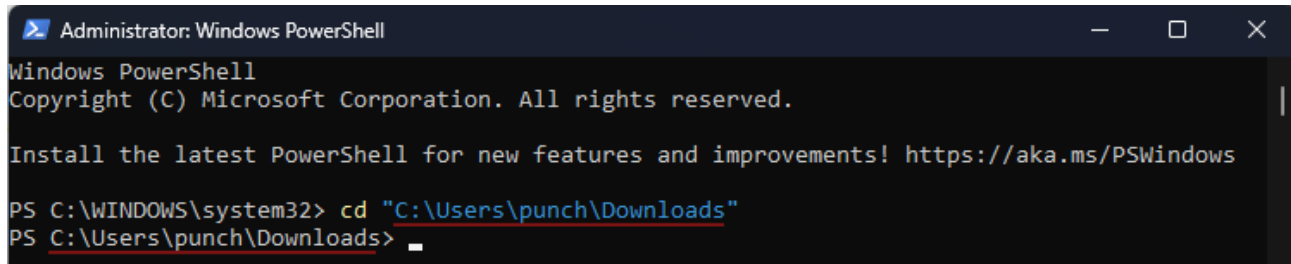


Figure 4. Example of how to change the working directory of the PowerShell Terminal

Now that you are in the same directory as the script, simply run the script as follows:

```
.\install_cpp_windows.ps1
```

At this point, the script will begin doing its “magic” and will install a couple of different things:

- 1) **Chocolatey** – a package manager that can automatically install various different software components. This will install the “choco” command.
- 2) **MinGW** – A windows-ported version of C++ binaries/executables to be able to develop and run C/C++ programs without any additional dependencies. This will install the “g++” command.
- 3) **VS Code** – A modern IDE (Integrated Development Environment), that supports developing programs in many different languages. This will install the “code” command.
- 4) **C++ Visual Studio Extensions** – A series of quality-of-life (QOL) extensions that make running and developing C++ programs more hassle-free.
- 5) **C/C++ Automatic Code Formatter** – Generates the .clang-format file used by VS Code to format C/C++ code. Formats code according to the style guide with respect to internal whitespace, brace-placement, and tabbing-rules.

Note that if any one of these already exists on your machine, the script will ignore setting up that tool and will not override the existing installation that you have of that software (with the exception of the .clang-format file).

3 Verifying Installation

This section will walk you through how to verify that your installation was successful.

3.1 Ensure Dependencies Installed

To Verify that the dependencies were installed properly, simply **run the program again**. If any of the statuses mentioned at the beginning of the program are in red or say “[Condition Not Met]”, then something went wrong with installing one of the dependencies.

```

Administrator: Windows PowerShell
PS C:\WINDOWS\system32> cd "C:/Users/punch/Downloads"
PS C:\Users\punch\Downloads> .\install_cpp_windows.ps1
Checking installation statuses...

-----
Chocolatey: [Condition Met]
MinGW (GCC Compiler): [Condition Met]
Visual Studio Code: [Condition Met]
VS Code C++ Extensions: [Condition Met]
-----


Chocolatey already installed.
MinGW already installed.
Visual Studio Code already installed.
C++ extensions already installed.
VS Code settings updated with C++ formatting rules.

C++ development environment setup is complete!
PS C:\Users\punch\Downloads>

```

Figure 5. Example output of the script upon a successful re-run of the script

3.2 Sample C++ Program Run

To test to make sure that the C++ binaries actually installed properly, first open up VS Code. The installation should have added a shortcut to VS Code to your desktop, but if that is not the case, then you can open VS Code by pressing  and then Typing in “Visual Studio Code”.

Once you are inside of VS Code, in the top-left, click on **File** > **Open Folder** and select a folder that contains some C++ code (preferably one of the programs that you have written for one of the labs).

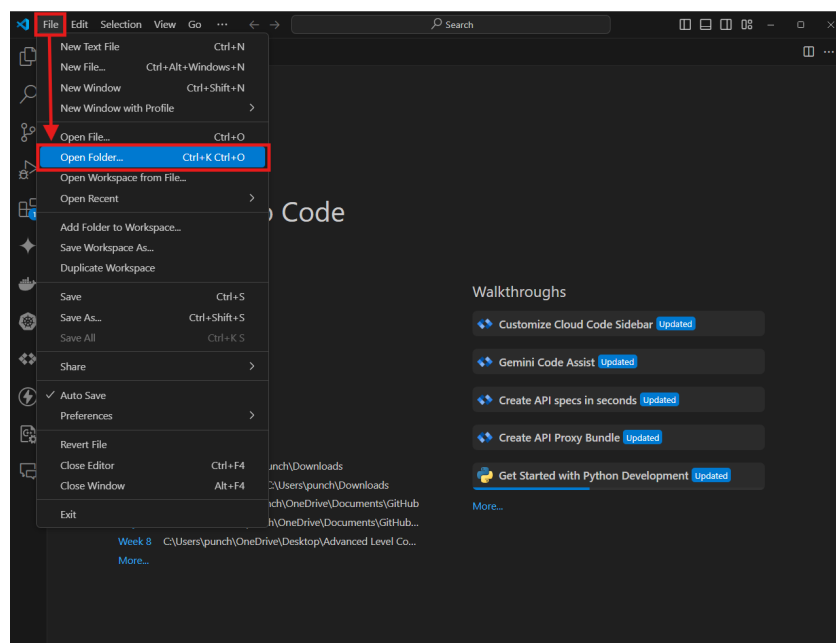
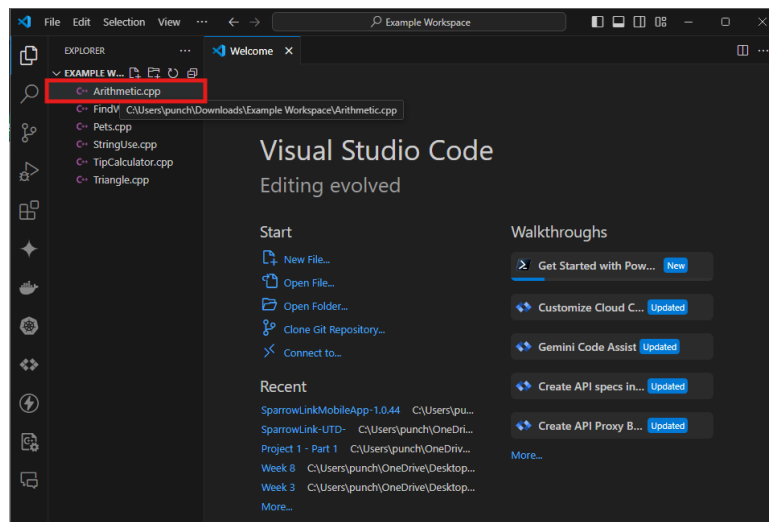
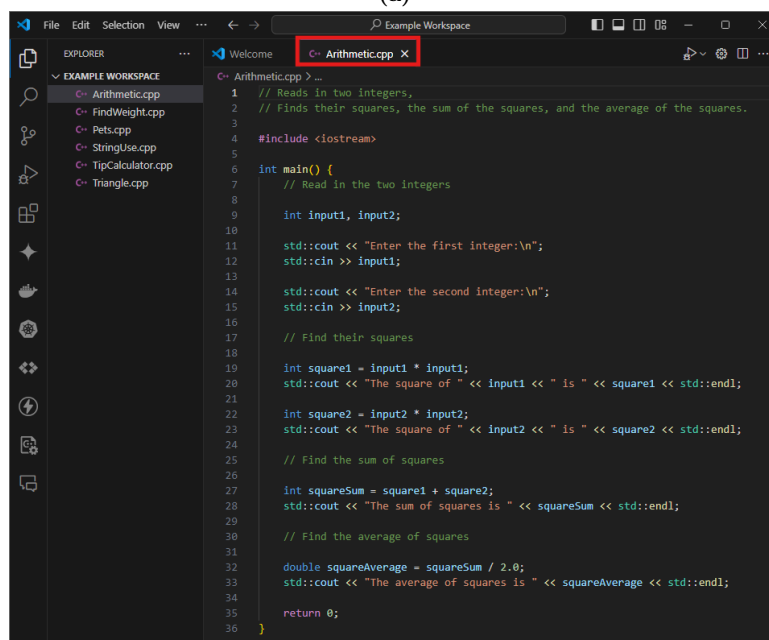


Figure 6. How to open a workspace in VS Code

Once selected, VS Code will open a new perspective with the project view on the left, and a place to display code on the right. Double click on one of your C++ source files on the left to open it up.




(a)



(b)

Figure 7. (a) Perspective before opening a file.
(b) Perspective after opening a file

Once the file is opened up, in the top right, click the run button (). This will prompt you to choose a configuration to run C/C++ code in. Select the option that lists g++. Do **NOT** select the one that mentions gcc.

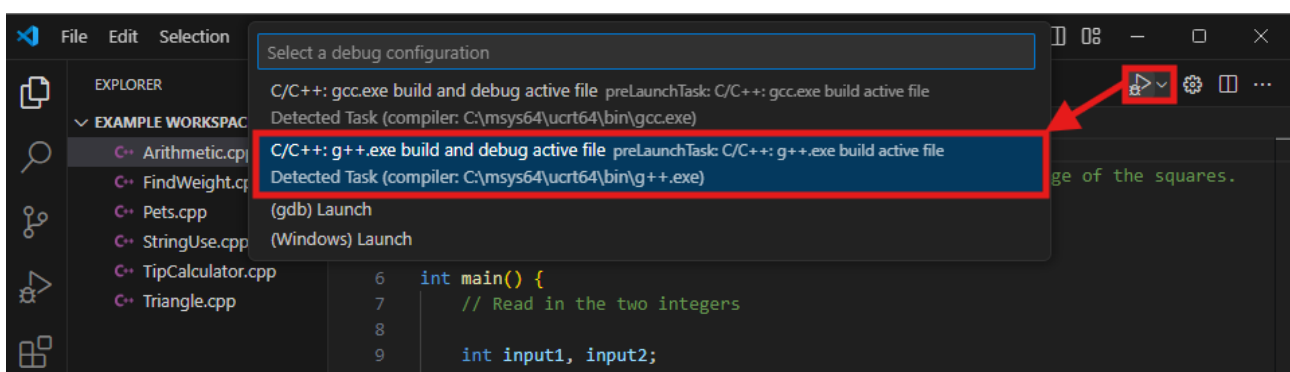


Figure 8. How to specify your C/C++ Compiler in VS Code

At this point, your code will first be compiled, and then ran. To view the code as it is being ran, click on the “Terminal” tab on your bottom hotbar. From here, you can interact with your program just as you would have using [Online GDB](#).

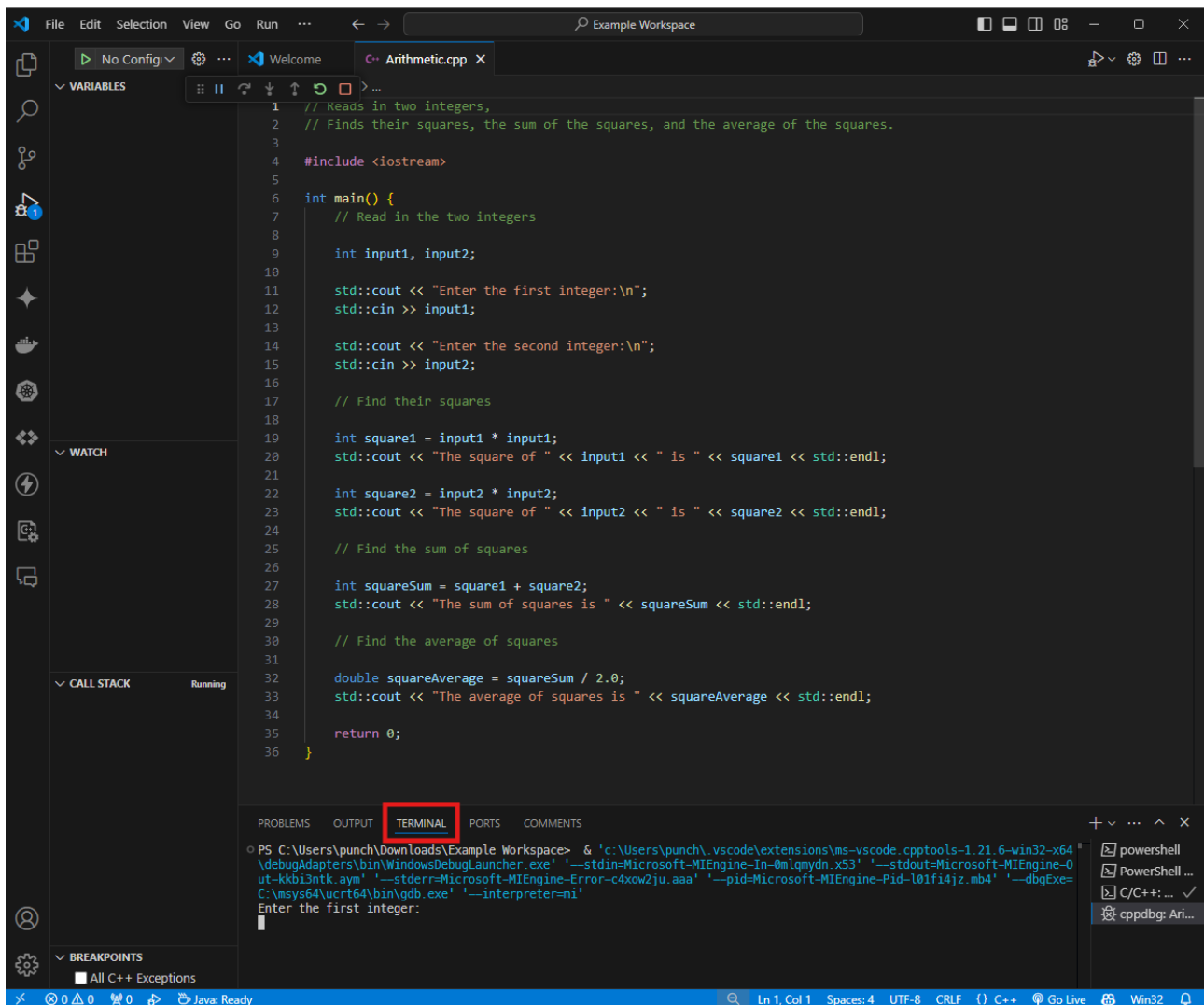


Figure 9. How to open the console for interactive programs

4 Closing Thoughts

Please note that, although your VS Code environment has been set up to automatically format your code upon saving a program, this is **not sufficient** to receive full marks for following the provided style guide. There are certain parts of the style guide’s specifications which this formatting does not account for, which you are expected to complete. Please pay careful attention to those things.

If you experience any difficulties getting your development environment set up, please see your TA before class or during the lab section.

5 Acknowledgement

The script and documentation provided to you was created and maintained by Matthew Sheldon, owner and original author. For any questions or updates, please contact me directly [via email](mailto:matthewtsheldon@gmail.com) (matthewtsheldon@gmail.com).