

## Visual Studio Code Introduction

Rev 1.1

### Overview:

In the Introduction to C Programming Introduction activity, you had a first look at a code editor when you used the Replit online tool. In this lesson you'll launch the Visual Studio Code application on your RPi system. This tool was installed and configured for you by the instructor. The instructor will first discuss the installation including required components and system settings. You'll then launch the tool, review the interface, write some initial code, and then run it. VS Code is the tool for this course. Practice with it to learn it well. As discussed in the earlier session, this is the tool of professional developers.

### Prerequisites:

Prior to beginning the instruction provided in this lesson you must have completed the following:

1. Linux Introduction

### Performance Outcomes:

1. Recognize the Visual Studio Code interface.
2. Identify requirements for VS Code to compile and execute C code.
3. Use the terminal to open a directory as a workspace in VS Code.
4. Use the VS Code Explore to create and access files in the workspace.
5. Use the VS Code editor to write and debug C code.
6. Use the Extensions collection in the Activity bar to view installed extensions.

### Resources:

1. [Getting started with Visual Studio Code](#)
2. [Download Visual Studio Code - Mac, Linux, Windows](#)
3. [How to fix slow Visual Studio Code on Raspberry Pi 4 - RATTICON](#)

### Materials:

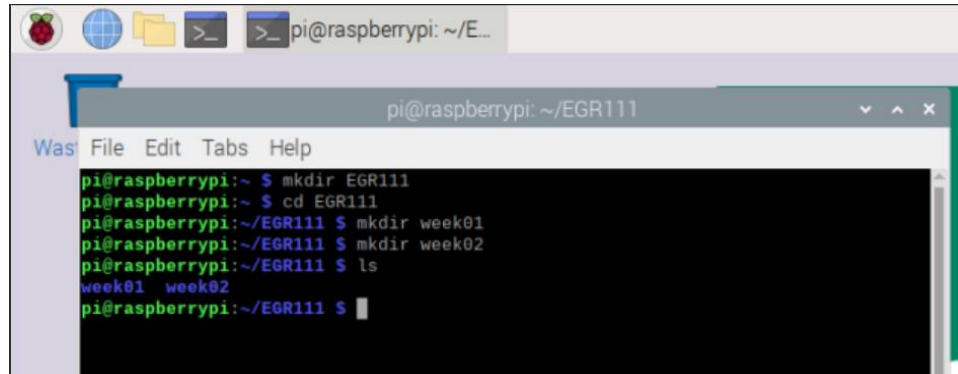
1. RPi System

### Directions:

1. Use the terminal commands learned in the Linux Introduction activity to create a workspace for this course. Open the terminal window and using the **mkdir** command to create an **EGR111** directory. You will use this workspace on the RPi system to complete the coding requirements for this class.

## EGR111 – Introduction to Computer Science (C Language)

2. Move to the EGR111 directory. Create a directory under this main directory named **Week01**. Create a directory under this main directory named **Week02**. List the content of the EGR111 workspace. Your terminal window should look similar to the image below.



```
pi@raspberrypi: ~/E...
pi@raspberrypi: ~/EGR111
Was: File Edit Tabs Help
pi@raspberrypi:~ $ mkdir EGR111
pi@raspberrypi:~ $ cd EGR111
pi@raspberrypi:~/EGR111 $ mkdir week01
pi@raspberrypi:~/EGR111 $ mkdir week02
pi@raspberrypi:~/EGR111 $ ls
week01 week02
pi@raspberrypi:~/EGR111 $
```

Figure 1 Terminal Window Showing Class Workspace Created

3. Open VS Code in your EGR111 workspace by entering the command **code .** with the dot after code indicating to open VS Code in the current directory.

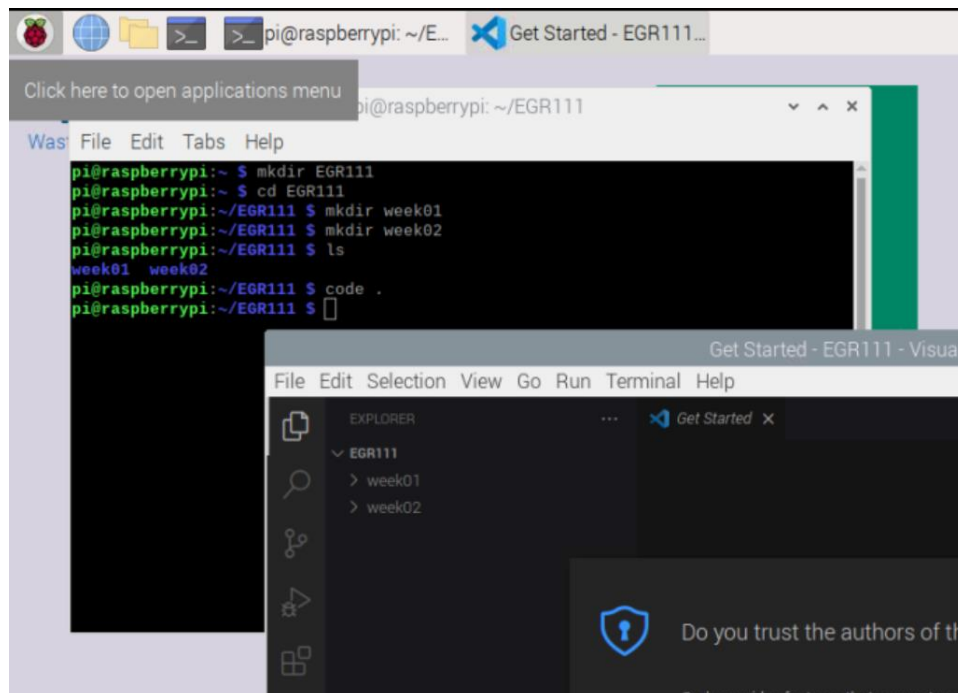


Figure 2 Visual Studio Code Opened on the Desktop using the code dot Command

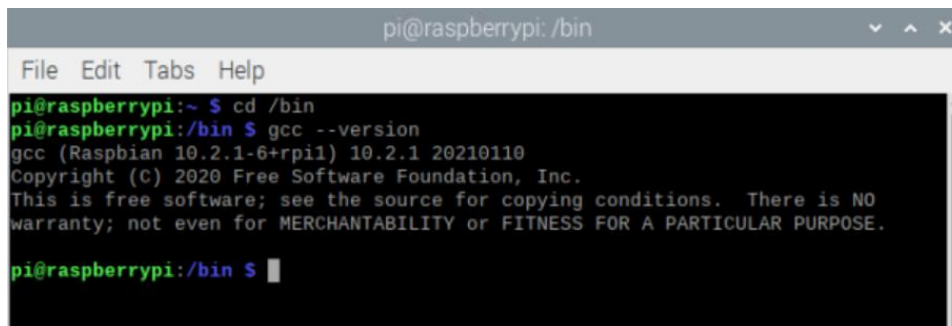
4. Verify that your directory structure and the VS Code window matches what the instructor is demonstrating. You must manage your files and directories in this class so that the required structure is created.

## EGR111 – Introduction to Computer Science (C Language)

In future lessons you will be pushing your code to a repository that is shared with the instructor. Your files and structures must match those specified in lessons and assignments.

5. The instructor will provide an overview of the Visual Studio Code IDE. For additional details or to review, view the [Getting started with Visual Studio Code](#) page and associated video.
6. The instructor will discuss the requirements for creating and debugging C code using VS Code on the Raspberry Pi.
  - a. A C compiler must be installed. The C compiler on your RPi unit is part of the Raspbian operating system and can be viewed using the following commands.

**gcc --version**



```
pi@raspberrypi: /bin
File Edit Tabs Help
pi@raspberrypi:~ $ cd /bin
pi@raspberrypi:/bin $ gcc --version
gcc (Raspbian 10.2.1-6+rpil) 10.2.1 20210110
Copyright (C) 2020 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
pi@raspberrypi:/bin $
```

*Figure 3 gcc compiler install in bin directory*

- a. An editor or IDE is required. As discussed in a prior lesson, VS Code is the most popular IDE. This can be installed from [Download Visual Studio Code - Mac, Linux, Windows](#) but a Linux package is available. Entering the following command on a Linux command prompt will install VS Code.

**sudo apt install code**

- b. VS Code provides editing and debugging for almost any language. To use VS Code to create C solutions an extension must be installed. Note that other extensions related to C programming are recommended. These were not installed when the class version of the OS was created. The intent was to keep the editor as “light” as possible. We may experiment with other extensions as we complete the class.

## EGR111 – Introduction to Computer Science (C Language)

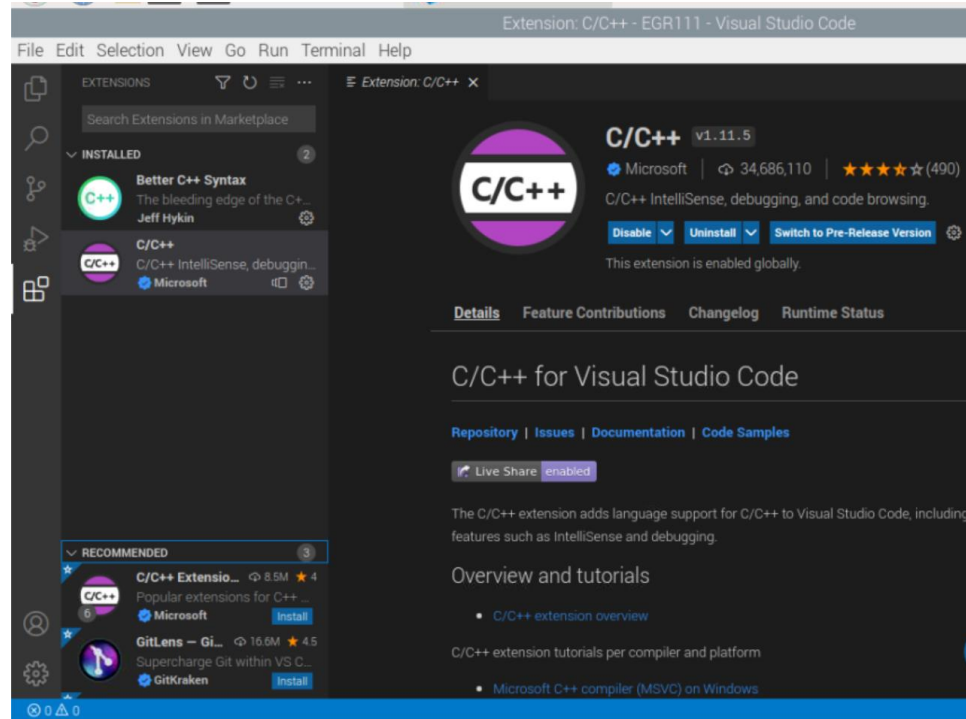


Figure 4 VS Code with C Extension Installed

7. Return to the Explorer tool on the Activity Bar and click on the week02 folder. We'll run our first C code – the perfunctory Hello World example.
8. Open an integrated terminal in VS Codes using the menu option and navigate to the week02 directory. Create a new file in the week02 directory named hello.c
9. Open the replit.com site on your RPi desktop. Copy and paste the hello world code from the prior introductory lesson into the hello.c file in VS Code.
10. The instructor will discuss each line of code.

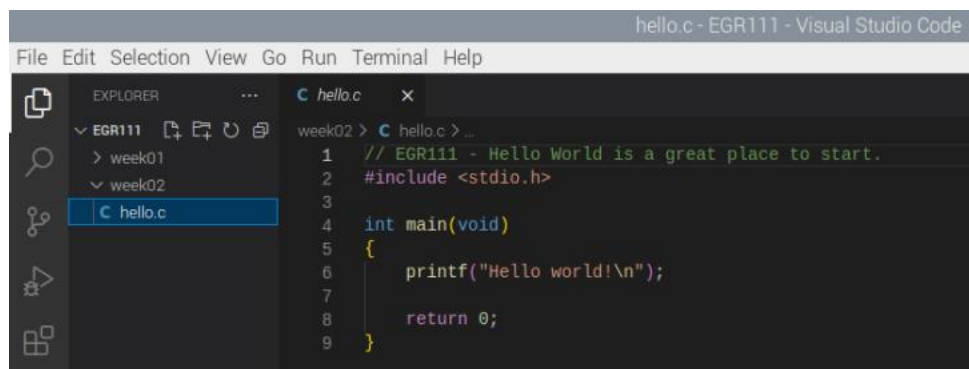


Figure 5 hello.c Open in the VS Code Editor

## EGR111 – Introduction to Computer Science (C Language)

11. To compile and run the sample code, select Start Debugging from the Run menu or press F5.

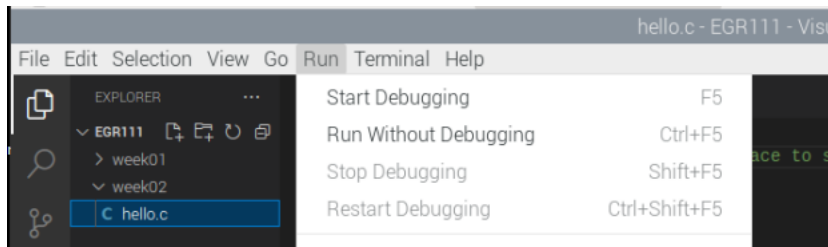


Figure 6 Running hello.c Sample Code

12. VS Code recognizes the .c source code and prompts you to confirm that you are building and running the active file. Select the first option as shown below. This task will be saved as a default operation for future debug sessions.

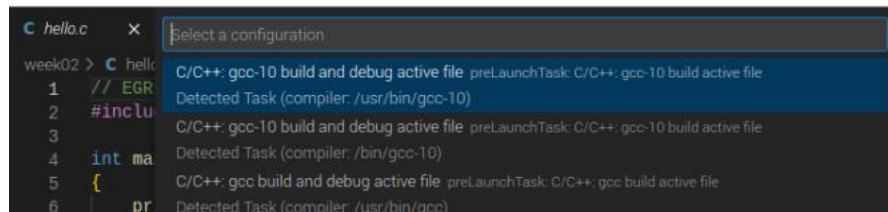


Figure 7 Selecting gcc Build Task

13. The .c source code is compiled into an executable and then runs in the integrated terminal. Note the output.

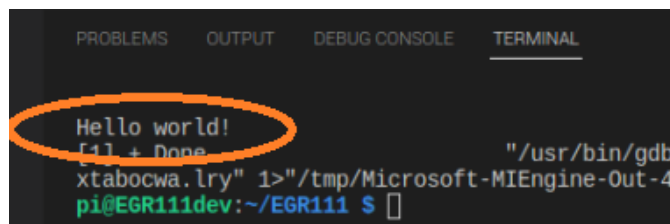


Figure 8 Hello.c Output Shown in the Integrated Terminal

14. Note that a VS Code workspace configuration hidden folder was created and contains a json file specifying the build tasks.

## EGR111 – Introduction to Computer Science (C Language)

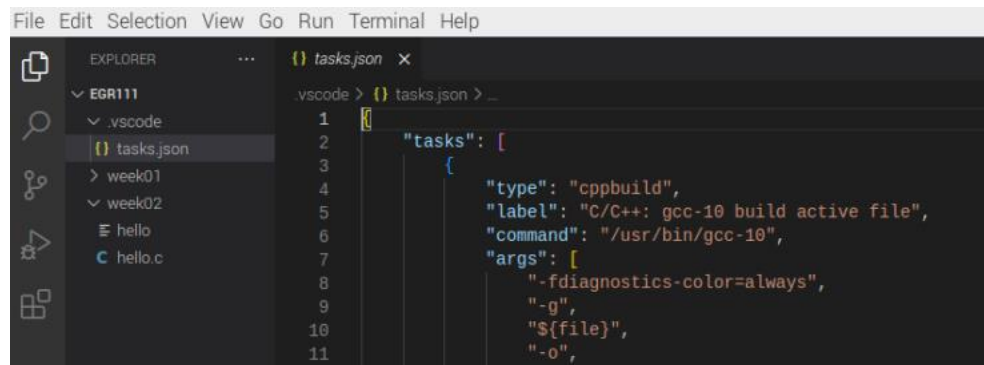


Figure 9 Build Tasks Defined in json File

15. Note also that the executable file *hello* has been created in the *week02* directory. The file can be selected but cannot be displayed in the editor. It is a binary file.

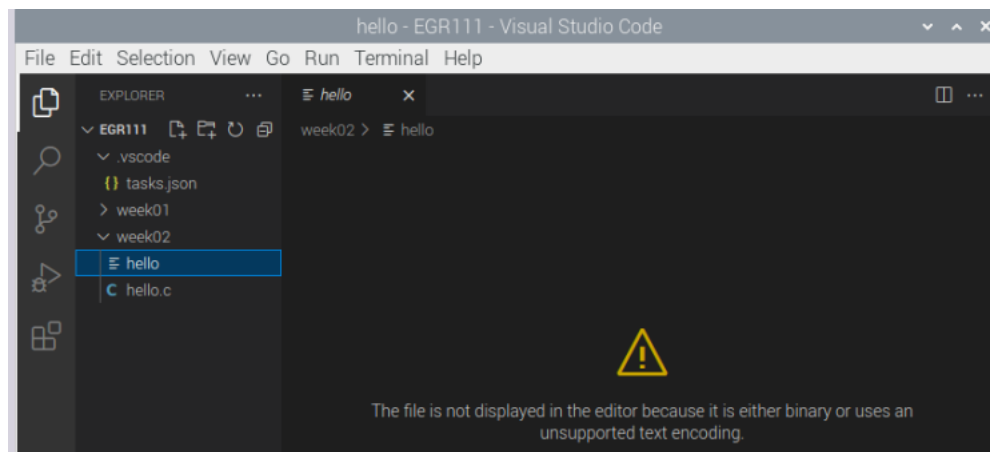


Figure 10 hello Executable Created During Build

16. View and run the executable from a bash shell terminal window.

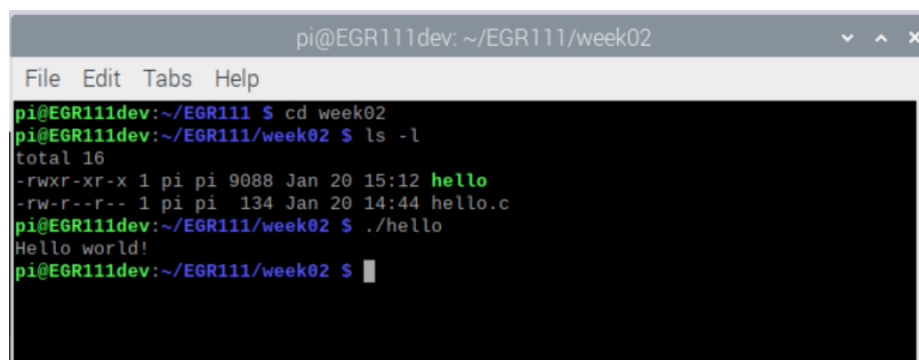


Figure 11 Enter ./hello to run the program at the bash command line

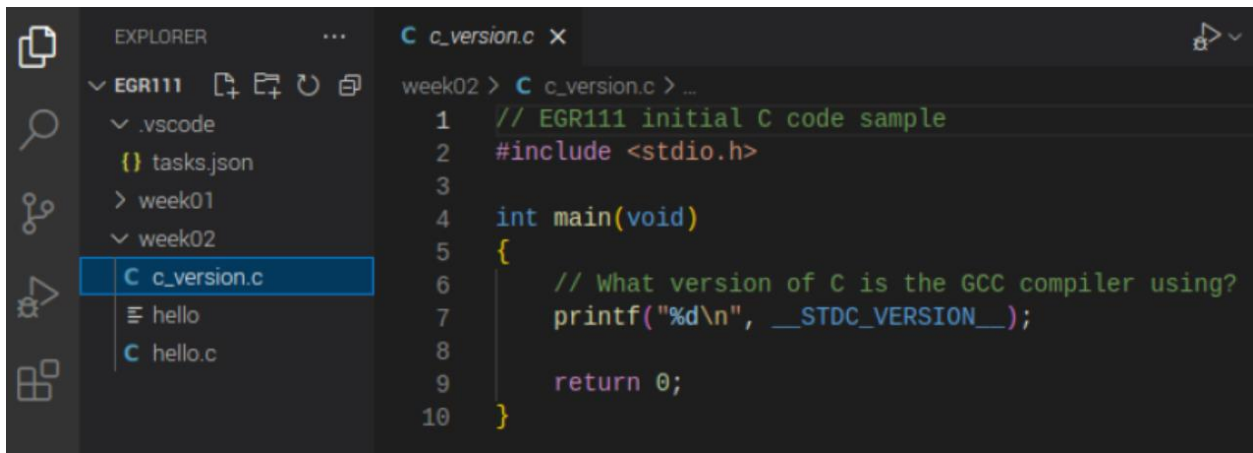
On you own...

## EGR111 – Introduction to Computer Science (C Language)

1. Open an integrated terminal in VS Codes and move to the week02 directory. Copy and paste the following wget command to download the c\_version.c example code.

```
wget https://k2controls.github.io/EGR111/lessons/res/c_version.c
```

2. Select the c\_version.c file in the Explorer to view the code.



```
1 // EGR111 initial C code sample
2 #include <stdio.h>
3
4 int main(void)
5 {
6     // What version of C is the GCC compiler using?
7     printf("%d\n", __STDC_VERSION__);
8
9     return 0;
10 }
```

3. Review each line and then press F5 to build and run. What version of C is being used? See the terminal pane for the output provided.
4. Search the Web for this version of C. Add a couple of comment lines to the code providing detail on this version. Save and close your revised c\_version.c code file.

### Assessments:

1. You will submit your EGR111 workspace to the instructor on or around week 03. Be sure your week02 directory contains both source and executable code for **hello** and **c\_version**.