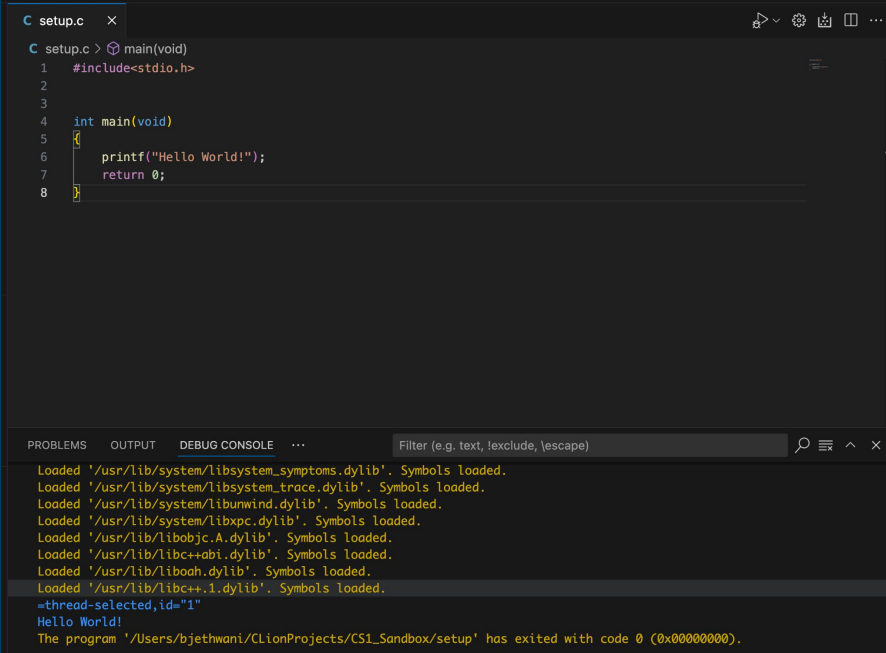


# How to set up and use GCC compiler on Visual Studio Code (Windows Only)



The screenshot shows the Visual Studio Code interface with a C file named `setup.c` open. The code in the editor is as follows:

```
C setup.c > main(void)
1 #include<stdio.h>
2
3
4 int main(void)
5 {
6     printf("Hello World!");
7     return 0;
8 }
```

Below the editor, the **DEBUG CONSOLE** tab is active, displaying the output of the program's execution:

```
Loaded '/usr/lib/system/libsystem_symptoms.dylib'. Symbols loaded.
Loaded '/usr/lib/system/libsystem_trace.dylib'. Symbols loaded.
Loaded '/usr/lib/system/libunwind.dylib'. Symbols loaded.
Loaded '/usr/lib/system/libxpc.dylib'. Symbols loaded.
Loaded '/usr/lib/libobjc.A.dylib'. Symbols loaded.
Loaded '/usr/lib/libc++abi.dylib'. Symbols loaded.
Loaded '/usr/lib/liboah.dylib'. Symbols loaded.
Loaded '/usr/lib/libc++1.dylib'. Symbols loaded.
=thread-selected,id="1"
Hello World!
The program '/Users/bjethwani/CLionProjects/CS1_Sandbox/setup' has exited with code 0 (0x00000000).
```

# Tutorial Introduction/Overview

## Goals/Objectives

- Show the steps to download a GCC compiler on VS Code.
- Show how to use the GCC compiler on VS Code.
- Show additional tools that come with the installation

## Requirements/Tools Needed

- Windows 8+ Device required
- VS Code Installed
- Google Chrome/Browser to install resources

# Table of Contents

<u><a href="#">GCC Background/Use</a></u> .....	4-5
<u><a href="#">Install GCC/VS Code Extensions</a></u> .....	6-7
<u><a href="#">Setting up GCC</a></u> .....	8-10
<u><a href="#">Trying/Using GCC</a></u> .....	11-14
<u><a href="#">Glossary</a></u> .....	15
<u><a href="#">What you learned</a></u> .....	16

# GCC History

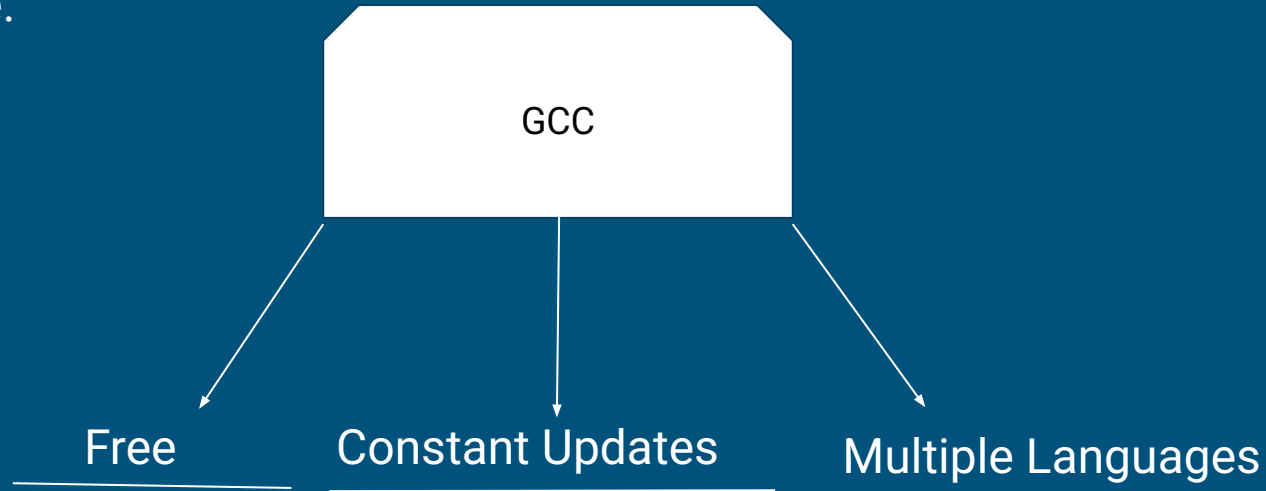
- GCC stands for the GNU Compiler Collection.
- GNU was the first free compiler created by non-profit open source methods.



# Why should you download the GCC compiler?

---

1. Without a compiler the code you type on a file cannot be processed.
2. GCC offers an easy to use debugger called GDB which allows you to debug your code.



# On Visual Studio Code Navigate to Extensions

EXTENSIONS: MARKETPLACE... ⌵

C/C++

CC

C/C++  
C/C++ IntelliSense, debugg...  
Microsoft

78.7M ★ 3.5

Install

C/C++ The...

CC

C/C++ Themes for C/C++ exten...

41.1M ★ 3.5

Install

C/C++ Runner

CC

Compile, run and debug ...  
franneck94

5.6M ★ 4.5

Install

C/C++ Com...

CC

Compile & Run single c/c++...  
danielpinto8zz6

4.5M ★ 4.5

Install

C/C++ Exten...

CC

Popular extensions for C++...

39M ★ 4.5

Install

Extension: C/C++ X

C/C++

Microsoft

78,742,855

★★★★☆ (577)

C/C++ IntelliSense, debugging, and code browsing.

Install

Auto Update

DETAILS

FEATURES

CHANGELOG

C/C++ for Visual Studio Code

Repository

Issues

Documentation

Code Samples

Live Share


enabled

The C/C++ extension adds language support for C/C++ to Visual Studio Code, including [editing \(IntelliSense\)](#) and [debugging](#) features.








Marketplace

Identifier	ms- vscode.cpptools
Version	1.23.6
Publish...	2016-03-29, 21:27:12
Last Released	2025-02-13, 19:49:35

Extension: C/C++



C/C++


Microsoft  [microsoft.com](https://microsoft.com) |  78,742,855 |      (577)

C/C++ IntelliSense, debugging, and code browsing.

Disable

Uninstall

Switch to Pre-Release Version

☒ Auto Update 


DETAILS

FEATURES

CHANGELOG

## C/C++ for Visual Studio Code

Repository | Issues | Documentation | Code Samples

 Live Share

enabled

The C/C++ extension adds language support for C/C++ to Visual Studio Code, including [editing \(IntelliSense\)](#) and [debugging](#) features.

Installation	
Identifier	ms-vscode.cpptools
Version	1.23.6
Last Updated	2025-02-20, 13:12:29
Size	218.39MB

## Installing the GCC Compiler

Note:

We will be downloading the GCC Compiler from [MSYS2](#)

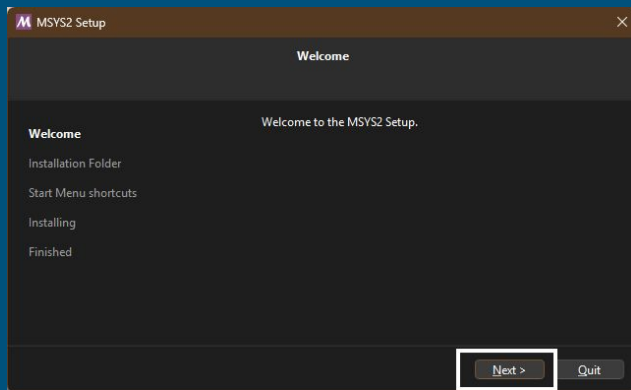
1. Download the link below

[https://github.com/msys2/msys2-installer/releases/download/2025-02-21/msys2-x86\\_64-20250221.exe](https://github.com/msys2/msys2-installer/releases/download/2025-02-21/msys2-x86_64-20250221.exe)

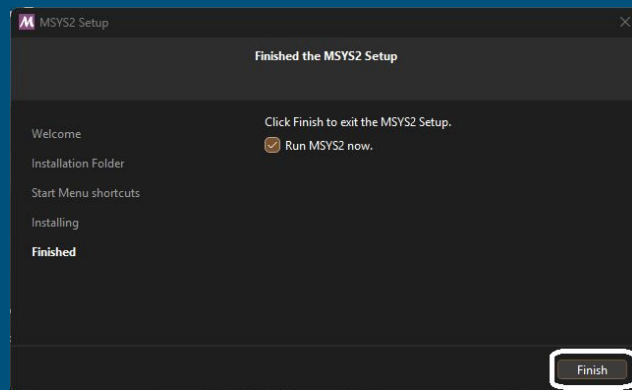
**Note:** This link will download the MinGW, which is a toolchain that contains GCC and various other tools/software. For more information visit this link:

<https://www.msys2.org/>

2. Run the installer and click next till finished



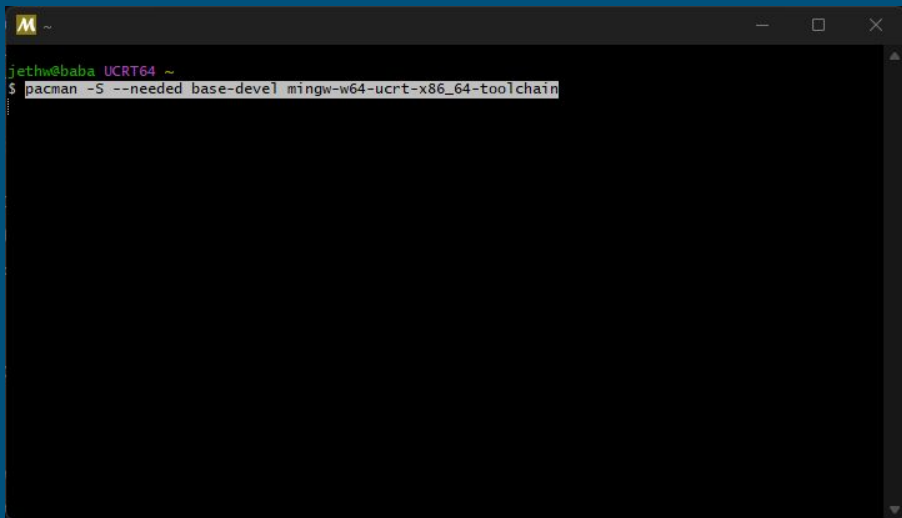
3. Click CMD and execute the following command



Accept default settings unless you want manual settings

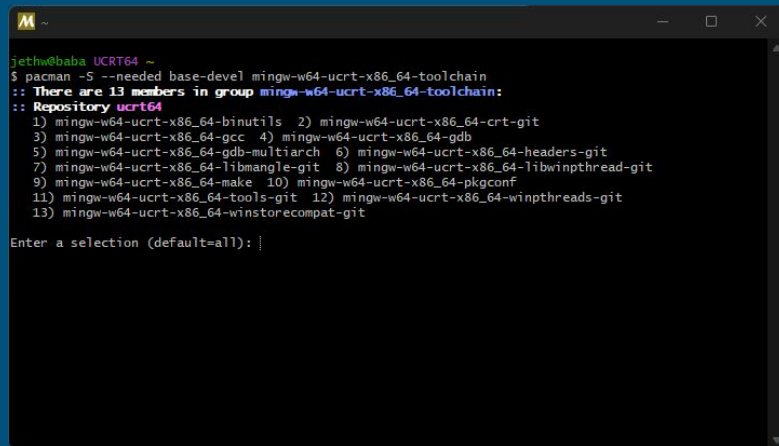
# Setting up GCC compiler

1. Copy and paste the following command:  
`pacman -S --needed base-devel mingw-w64-ucrt-x86_64-toolchain`



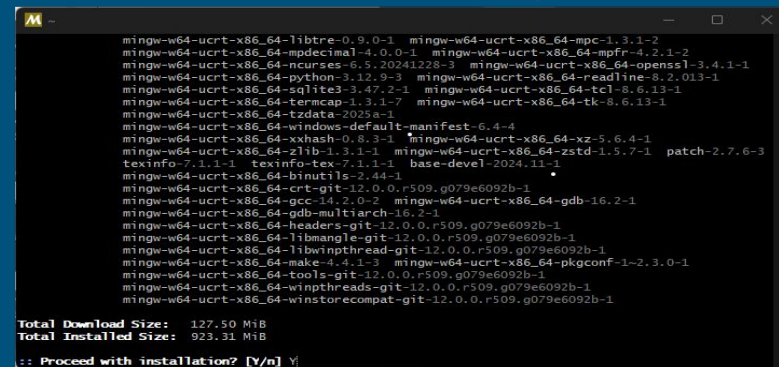
```
jethw@baba UCRT64 ~  
$ pacman -S --needed base-devel mingw-w64-ucrt-x86_64-toolchain
```

2. Hit enter to download all default tools



```
jethw@baba UCRT64 ~  
$ pacman -S --needed base-devel mingw-w64-ucrt-x86_64-toolchain  
:: There are 13 members in group mingw-w64-ucrt-x86_64-toolchain:  
:: Repository ucrt64  
1) mingw-w64-ucrt-x86_64-binutils 2) mingw-w64-ucrt-x86_64-crt-git  
3) mingw-w64-ucrt-x86_64-gcc 4) mingw-w64-ucrt-x86_64-gdb  
5) mingw-w64-ucrt-x86_64-gdb-multiarch 6) mingw-w64-ucrt-x86_64-headers-git  
7) mingw-w64-ucrt-x86_64-libmangle-git 8) mingw-w64-ucrt-x86_64-libwinpthread-git  
9) mingw-w64-ucrt-x86_64-make 10) mingw-w64-ucrt-x86_64-pkgconf  
11) mingw-w64-ucrt-x86_64-tools-git 12) mingw-w64-ucrt-x86_64-winthreads-git  
13) mingw-w64-ucrt-x86_64-winstorecompat-git  
Enter a selection (default=all): |
```

3. Type Y to proceed with installation



```
mingw-w64-ucrt-x86_64-libtbb-0.9.0-1 mingw-w64-ucrt-x86_64-mpc-1.3.1-2  
mingw-w64-ucrt-x86_64-mpdecimal-4.0.0-1 mingw-w64-ucrt-x86_64-mpfr-4.2.1-2  
mingw-w64-ucrt-x86_64-ncurses-6.5.20241228-3 mingw-w64-ucrt-x86_64-openssl-3.4.1-1  
mingw-w64-ucrt-x86_64-python-3.12.9-3 mingw-w64-ucrt-x86_64-readline-8.2.013-1  
mingw-w64-ucrt-x86_64-qclite3-3.47.2-1 mingw-w64-ucrt-x86_64-tcl-8.6.13-1  
mingw-w64-ucrt-x86_64-termcap-1.3.1-7 mingw-w64-ucrt-x86_64-tk-8.6.13-1  
mingw-w64-ucrt-x86_64-tzdata-2025a-1  
mingw-w64-ucrt-x86_64-windows-default-manifest-6.4-4  
mingw-w64-ucrt-x86_64-xxhash-0.8.3-1 mingw-w64-ucrt-x86_64-xxz-5.6.4-1  
mingw-w64-ucrt-x86_64-zlib-1.3.1-1 mingw-w64-ucrt-x86_64-zstd-1.5.7-1 patch-2.7.6-3  
texinfo-7.1.1-1 texinfo-tex-7.1.1-1 base-devel-2024.11.1  
mingw-w64-ucrt-x86_64-binutils-2.44-1  
mingw-w64-ucrt-x86_64-crt-git-12.0.0.r509.g079e6092b-1  
mingw-w64-ucrt-x86_64-gcc-14.2.0-2 mingw-w64-ucrt-x86_64-gdb-16.2-1  
mingw-w64-ucrt-x86_64-gdb-multiarch-16.2-1  
mingw-w64-ucrt-x86_64-headers-git-12.0.0.r509.g079e6092b-1  
mingw-w64-ucrt-x86_64-libmangle-git-12.0.0.r509.g079e6092b-1  
mingw-w64-ucrt-x86_64-libwinpthread-git-12.0.0.r509.g079e6092b-1  
mingw-w64-ucrt-x86_64-make-4.4.1-3 mingw-w64-ucrt-x86_64-pkgconf-1-2.3.0-1  
mingw-w64-ucrt-x86_64-tools-git-12.0.0.r509.g079e6092b-1  
mingw-w64-ucrt-x86_64-winthreads-git-12.0.0.r509.g079e6092b-1  
mingw-w64-ucrt-x86_64-winstorecompat-git-12.0.0.r509.g079e6092b-1  
Total Download Size: 127.50 MiB  
Total Installed Size: 923.31 MiB  
:: Proceed with installation? [Y/n] Y
```



# Finalizing GCC Installation Part 1

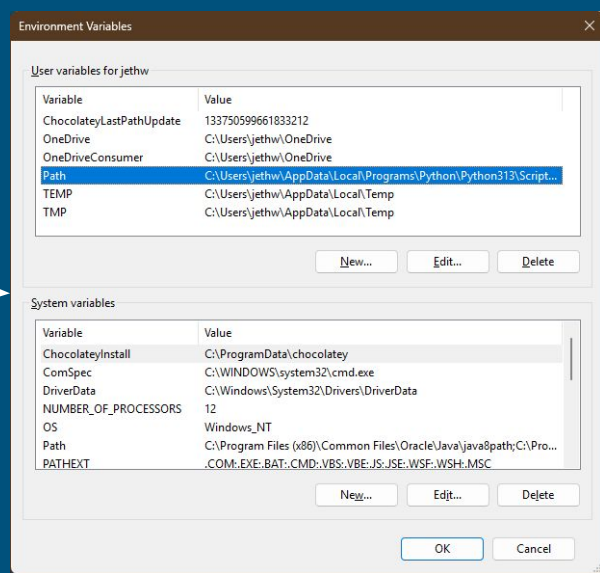
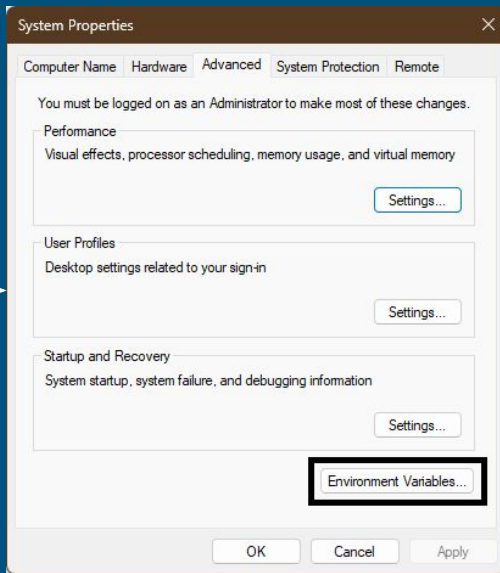
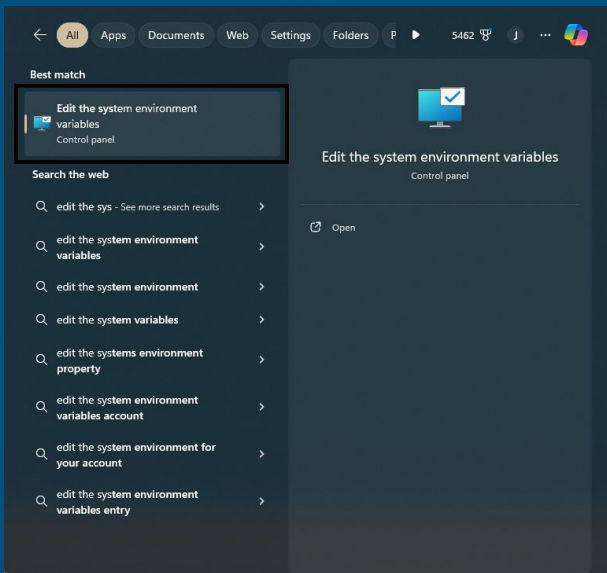
Now that all necessary components have been downloaded we need to make our system can find and recognize our compiler. We will do this by completing the first part:

**Note:** Changing user path changes for user rather than system. Changes won't apply for other users.

1. In your windows search, type: "Edit the system environment variables"

2. Click on the selection and then click on the option environment variables.

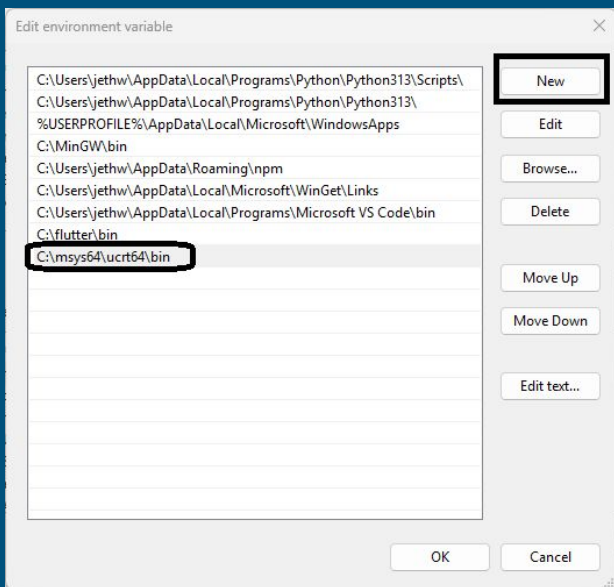
3. Find the variable Path under user and click edit.



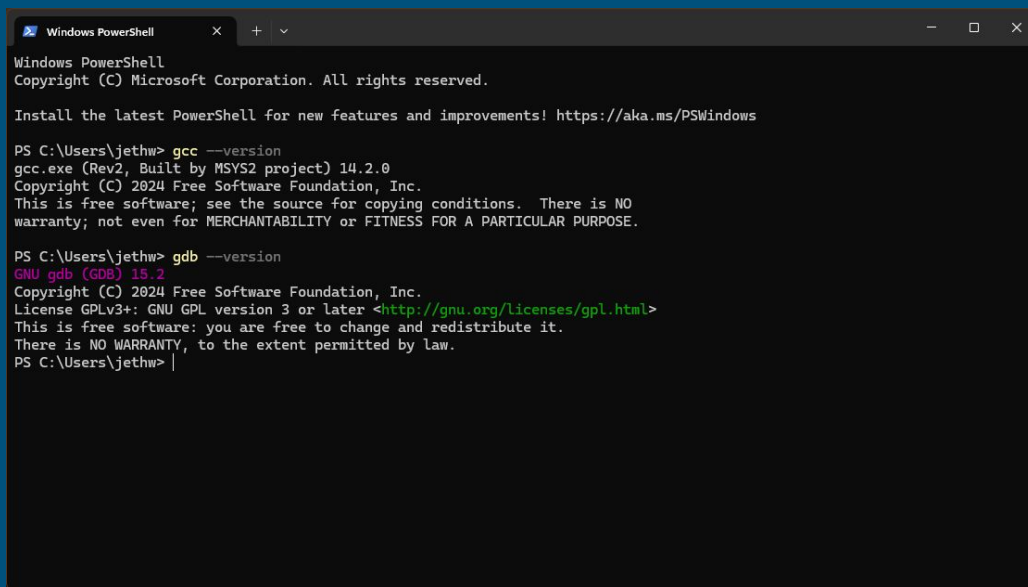
## Finalizing GCC Installation Part 2

We will configure our \$PATH variable to detect the bin folder of our MinGW download which contains our GCC Compiler. To do this we have to edit our PATH environment variable.

1. Click New and type the following path if you accepted all default settings on installation: C:\mysys64\ucrt64\bin. Otherwise, type in the relative path of the directory you saved it too.



2. Click OK until you exit the control panel. Now to make sure that the tools are downloaded correctly and your system can detect them, run the following commands in your command prompt:

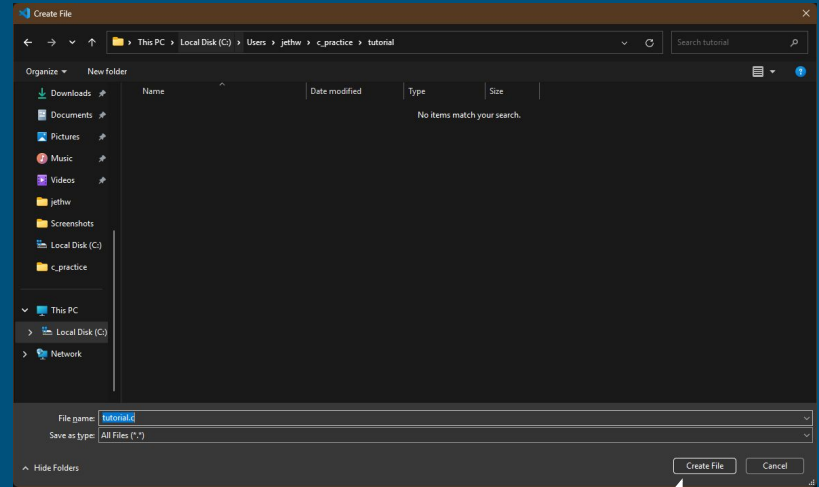
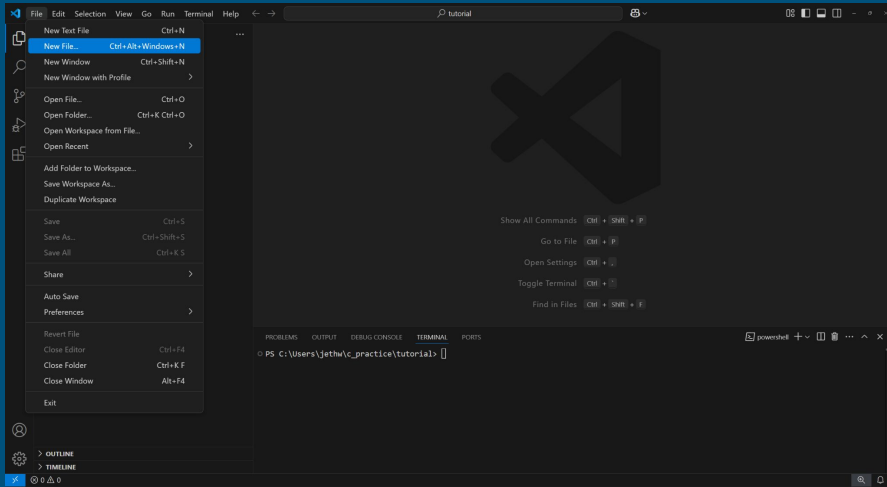


# Using GCC with Visual Studio Code Part 1

Now that we have verified our installation of GCC on our system, we can now use it on our desired platform VS Code. Let's create a C file!

1. In VS Code click file and click the new file option and name it "insert name here".c

2. Now choose a location for where you want to save the file and click create file.



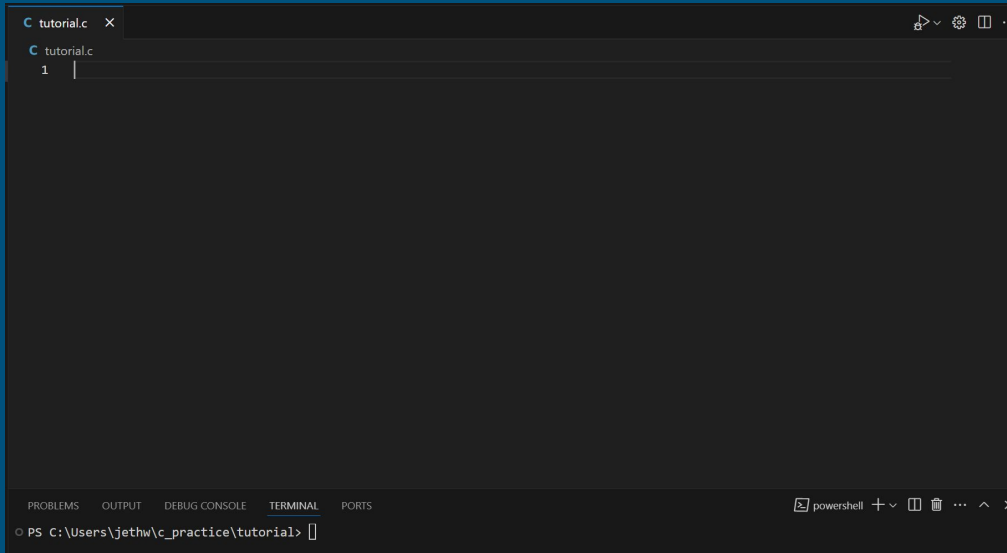
**Note:** Do not include the quotation marks around your chosen name

## Using GCC with Visual Studio Code Part 2

Now we will actually write some sample code on the new file which you created. Make sure to type the code letter for letter!

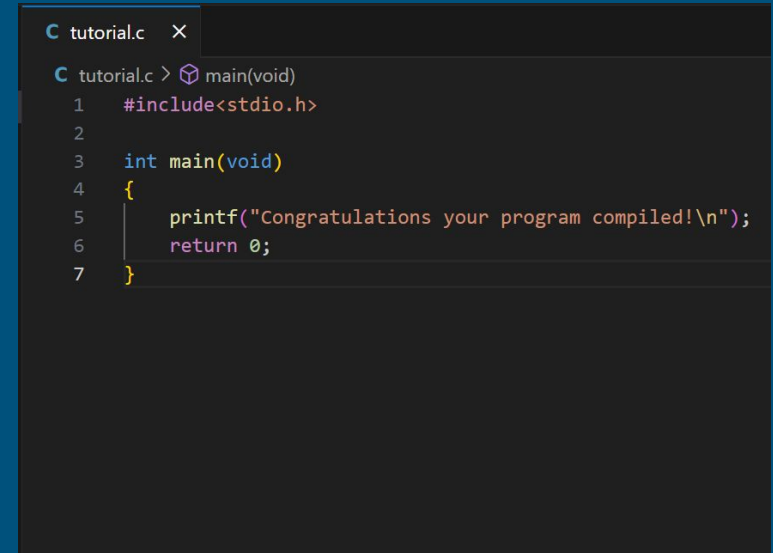
1. After creating the file, you should be brought to a screen which has the file open in the VS code editor.

2. Now type the following program code in the file.



```
C tutorial.c x
C tutorial.c
1 |

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\Users\jethw\c_practice\tutorial>
```

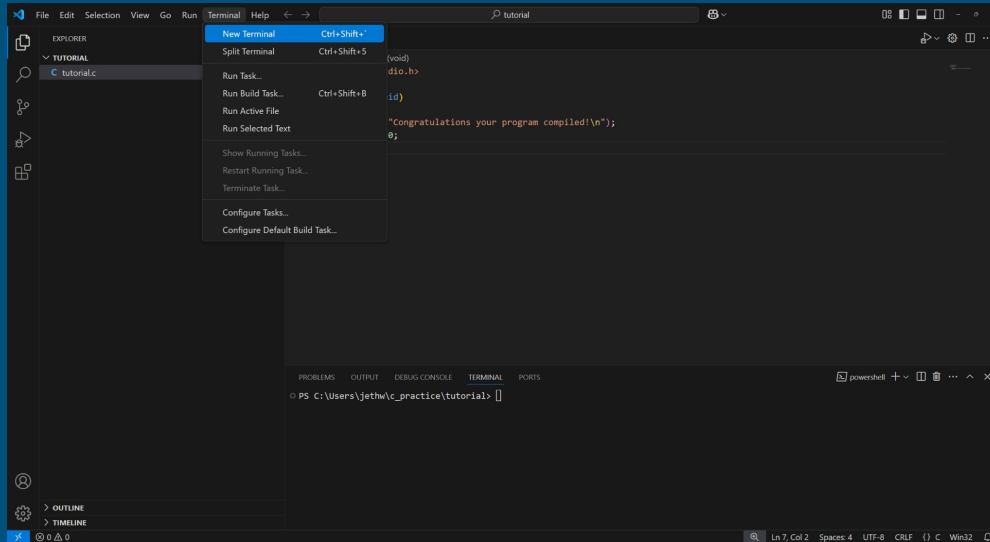


```
C tutorial.c x
C tutorial.c > main(void)
1  #include<stdio.h>
2
3  int main(void)
4  {
5      printf("Congratulations your program compiled!\n");
6      return 0;
7  }
```

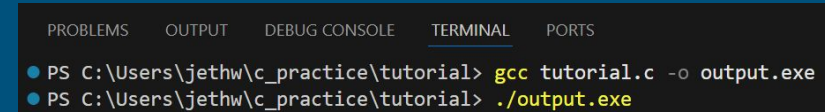
# Using GCC with Visual Studio Code Part 3

At last, we will now compile our C file which has executable code on it!

1. Select the terminal tab on the bottom of your screen. If it's not there by default then click terminal on the top of your screen, and click new terminal.



2. Now type the following commands in the terminal to compile and execute your file.



**Note:** After gcc make sure to put your "file name".c and not tutorial.c if you haven't named your file that.

The screenshot shows the Visual Studio Code interface with a C program named `tutorial.c` open in the editor. The Explorer sidebar on the left shows the project structure with `output.exe` and `tutorial.c`. The editor displays the following code:

```
C tutorial.c > main(void)
1  #include<stdio.h>
2
3  int main(void)
4  {
5      printf("Congratulations your program compiled!\n");
6      return 0;
7  }
```

At the bottom, the TERMINAL panel shows the execution of the program:

```
PS C:\Users\jethw\c_practice\tutorial> gcc tutorial.c -o output.exe
PS C:\Users\jethw\c_practice\tutorial> ./output.exe
Congratulations your program compiled!
PS C:\Users\jethw\c_practice\tutorial>
```

The status bar at the bottom indicates the cursor is at line 7, column 2, with 4 spaces, UTF-8 encoding, CRLF line endings, and a C file type.

Finally, your screen should look like this!

**Note:** Your program output is placed right under the terminal commands.

# Glossary

---

**Compiler:** Software which translates a programming language/code into machine code understandable by the computer.

**CMD(Command Prompt):** Interface/program which allows you to interact with the operating system using specific commands.

**Debugger:** Software tool which allows programmers to track and see the state of their program and it's variables.

**\$PATH:** A variable in your system which contains the paths for different applications "bins". These paths tell the operating system where to execute the application files.

**DIRECTORY:** A folder on your system located in a specific location in your operating system. Used for storing files, both for applications and personal use.

**BIN:** Directory which contains the files to be executed by the operating system so that the application can be run.

# Conclusion

---

In this tutorial you were able to do the following:

- Download GCC and set it up with VS code.
- Create a C file in VS code for compilation.
- Write a C program and compile it successfully.

Additional resources:

- <https://www.gnu.org/home.en.html>
- <https://gcc.gnu.org/>
- <https://github.com/gcc-mirror/gcc>