



Windows 配置VSCode

Visual Studio Code（简称VSCode）是由微软开发的一款免费、开源的代码编辑器。它支持多种编程语言，并且可以通过安装扩展来增加对其他语言和工具的支持。VS Code以其轻量级、高性能和强大的功能而受到开发者的广泛欢迎。它具有以下几个特点：

- **多语言支持**：VS Code支持C、C++、Python、PHP、JavaScript、TypeScript、Rust等编程语言，并且可以通过安装扩展来支持更多语言。
- **代码调试**：内置了调试功能，支持断点、步进、变量观察等，并且可以轻松地与多种运行时和语言进行集成。
- **扩展市场**：拥有一个庞大的扩展市场，用户可以根据需要安装扩展来增加新功能或支持新语言。
- **多平台支持**：VS Code可以在Windows、macOS和Linux操作系统上运行。

Step 1：安装MinGW

C/C++是一门高级语言，想要运行手中的代码，我们必须使用C/C++编译器将C/C++语言写出的代码文件转换为机器语言，这样计算机才能运行程序。

- **GCC (GNU Compiler Collection)**：由GNU项目开发，支持多种编程语言，包括C和C++，它是最流行的开源编译器之一，主要用于Linux，FreeBSD等类Unix系统，在Windows系统上则有GCC的windows版——MinGW（Minimalist GNU for Windows）
- **Clang**：由Apple公司开发，基于LLVM编译器基础设施。Clang支持C、C++、Objective-C等语言，以其出色的诊断信息和模块化设计而闻名
- **MSVC(Microsoft Visual C++)**：微软Visual Studio集成开发环境的一部分，专门用于Windows平台上的C++、C和汇编语言开发。

目前C/C++编译器有很多，主流的有：

本教程我们安装的是MinGW，MinGW也是GCC，可以理解为Windows版的GCC。

下载MinGW

镜像下载

镜像下载地址: http://202.112.113.225/winlibs-x86_64-posix-seh-gcc-14.2.0-llvm-18.1.8-mingw-w64ucrt-12.0.0-r1.zip

为了提高下载速度, 我提供了镜像, 访问以上链接必须连接校园网

官方下载

官方下载地址: <https://winlibs.com/#download-release>

Release versions

UCRT runtime



点击这个链接下载

- GCC 14.2.0 (with **POSIX** threads) + LLVM/Clang/LLD/LLDB 18.1.8 + MinGW-w64 12.0.0 UCRT - release 1 **(LATEST)**
 - Win32: [7-Zip archive*](#) | [Zip archive](#) - without LLVM/Clang/LLD/LLDB: [7-Zip archive*](#) | [Zip archive](#)
 - Win64: [7-Zip archive*](#) | [Zip archive](#) - without LLVM/Clang/LLD/LLDB: [7-Zip archive*](#) | [Zip archive](#)
- GCC 14.2.0 (with **MCF** threads) + MinGW-w64 12.0.0 UCRT (release 1)
 - Win32: [7-Zip archive*](#) | [Zip archive](#)
 - Win64: [7-Zip archive*](#) | [Zip archive](#)
- GCC 14.1.0 (with **POSIX** threads) + LLVM/Clang/LLD/LLDB 18.1.8 + MinGW-w64 12.0.0 (UCRT) - release 3
 - Win32: [7-Zip archive*](#) | [Zip archive](#) - without LLVM/Clang/LLD/LLDB: [7-Zip archive*](#) | [Zip archive](#)
 - Win64: [7-Zip archive*](#) | [Zip archive](#) - without LLVM/Clang/LLD/LLDB: [7-Zip archive*](#) | [Zip archive](#)
- GCC 14.1.0 (with **MCF** threads) + MinGW-w64 11.0.1 (UCRT) - release 1
 - Win32 (without LLVM/Clang/LLD/LLDB): [7-Zip archive*](#) | [Zip archive](#)
 - Win64 (without LLVM/Clang/LLD/LLDB): [7-Zip archive*](#) | [Zip archive](#)
- GCC 13.3.0 (with **POSIX** threads) + MinGW-w64 11.0.1 (UCRT) - release 1
 - Win32 (without LLVM/Clang/LLD/LLDB): [7-Zip archive*](#) | [Zip archive](#)
 - Win64 (without LLVM/Clang/LLD/LLDB): [7-Zip archive*](#) | [Zip archive](#)

速度可能比较慢

安装MinGW

解压MinGW

 winlibs-i686-posix-dwarf-gcc-14.2.0-llvm-18.1.8-mingw-w64ucrt-12.0.0-r1.zip
 winlibs-i686-posix-dwarf-gcc-14.2.0-llvm-18.1.8-mingw-w64ucrt-12.0.0-r1

下载好的MinGW是一个zip压缩包, 将其解压到你放到的任何目录, 例如这里, 我将解压出的文件夹放到了C:\Users\ziyang\Downloads\winlibs-i686-posix-dwarf-gcc-14.2.0-llvm-18.1.8-mingw-w64ucrt-12.0.0-r1

配置环境变量

可执行程序

简单来说, 可以运行的程序是一个可执行程序。我们在计算机上常用的微信, 浏览器等都是可执行程序, 在Windows平台上, 这些可执行程序都有.exe后缀。同样的, 编译器也是一个可执行程序, 如下图所示, 我们后面要使用的gcc编译器也是一个可执行程序 (gcc.exe)

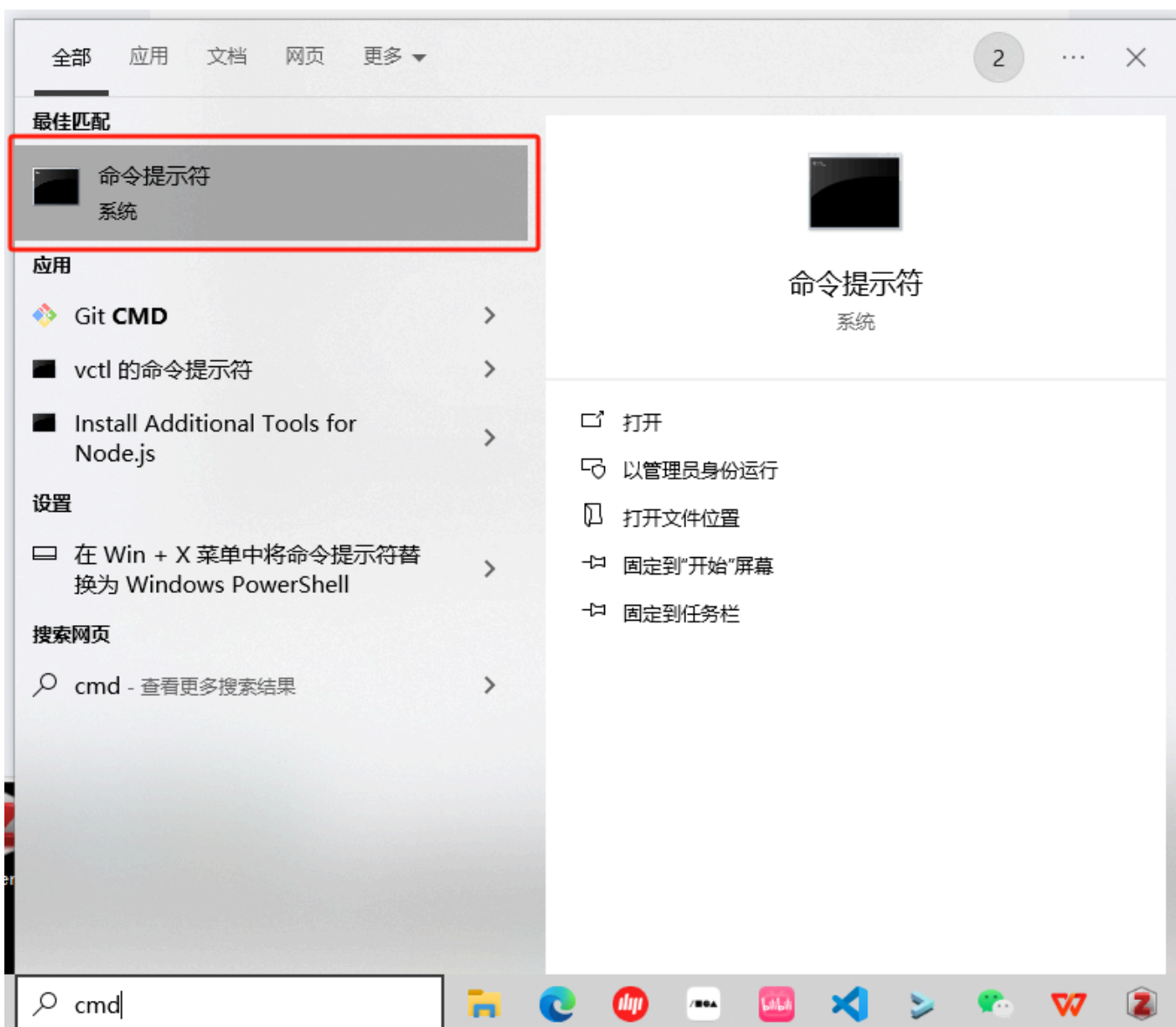
FileCheck.exe
find-all-symbols.exe
g++.exe
gcc.exe
gcc-ar.exe
gcc-nm.exe

图形用户界面与命令行界面

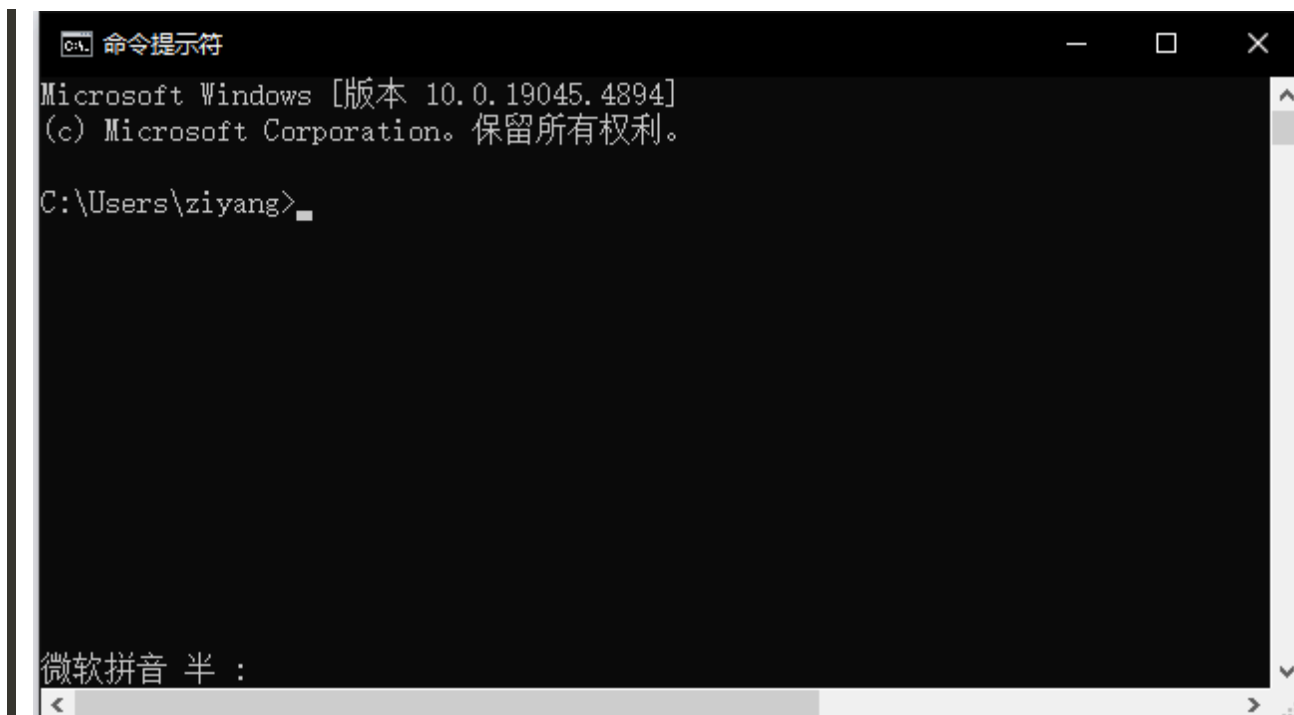
图形用户界面（Graphical User Interface，简称GUI）和命令行界面（Command Line Interface，简称CLI）是人与机器交互的方式。现如今我们常用的是GUI，操作系统提供了丰富的图形界面（如窗口、图标、菜单、工具栏等）来与计算机程序进行交互。而在GUI出现之前，人们都是用CLI与计算机进行交互，它允许用户通过输入文本命令来执行各种操作和程序。

CMD

CMD是Windows操作系统中的命令行界面，可以在搜索框中搜索CMD打开



打开后的界面如下，我们可以在该界面中输入命令



环境变量

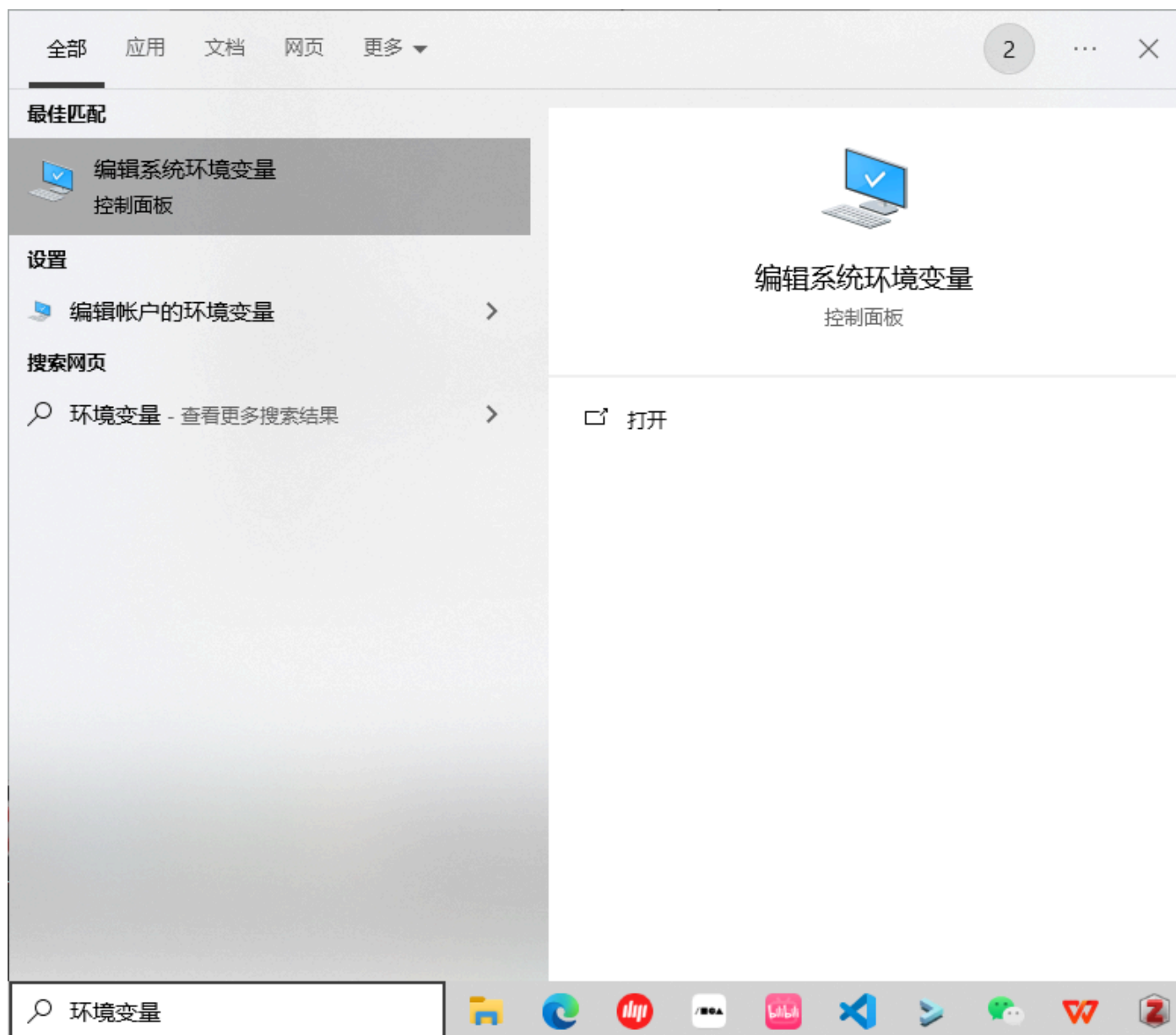
在Windows操作系统中，环境变量是一种存储系统或用户信息的机制，这些信息可以被操作系统、程序或脚本访问和使用。环境变量通常用于指定文件路径、系统配置、用户偏好设置等。环境变量以一个键值对的方式保存信息。

其中如果想要在当你在命令行中输入`gcc --version`并按Enter键后，命令行会找PATH这个环境变量，然后遍历其指定的所有目录寻找`gcc.exe`这个程序，如果能找到就执行，无法找到则会爆出错误：



因此为了系统能够找到我们下载的gcc编译器，我们需要将其加入到PATH环境变量中，添加的方法如下：

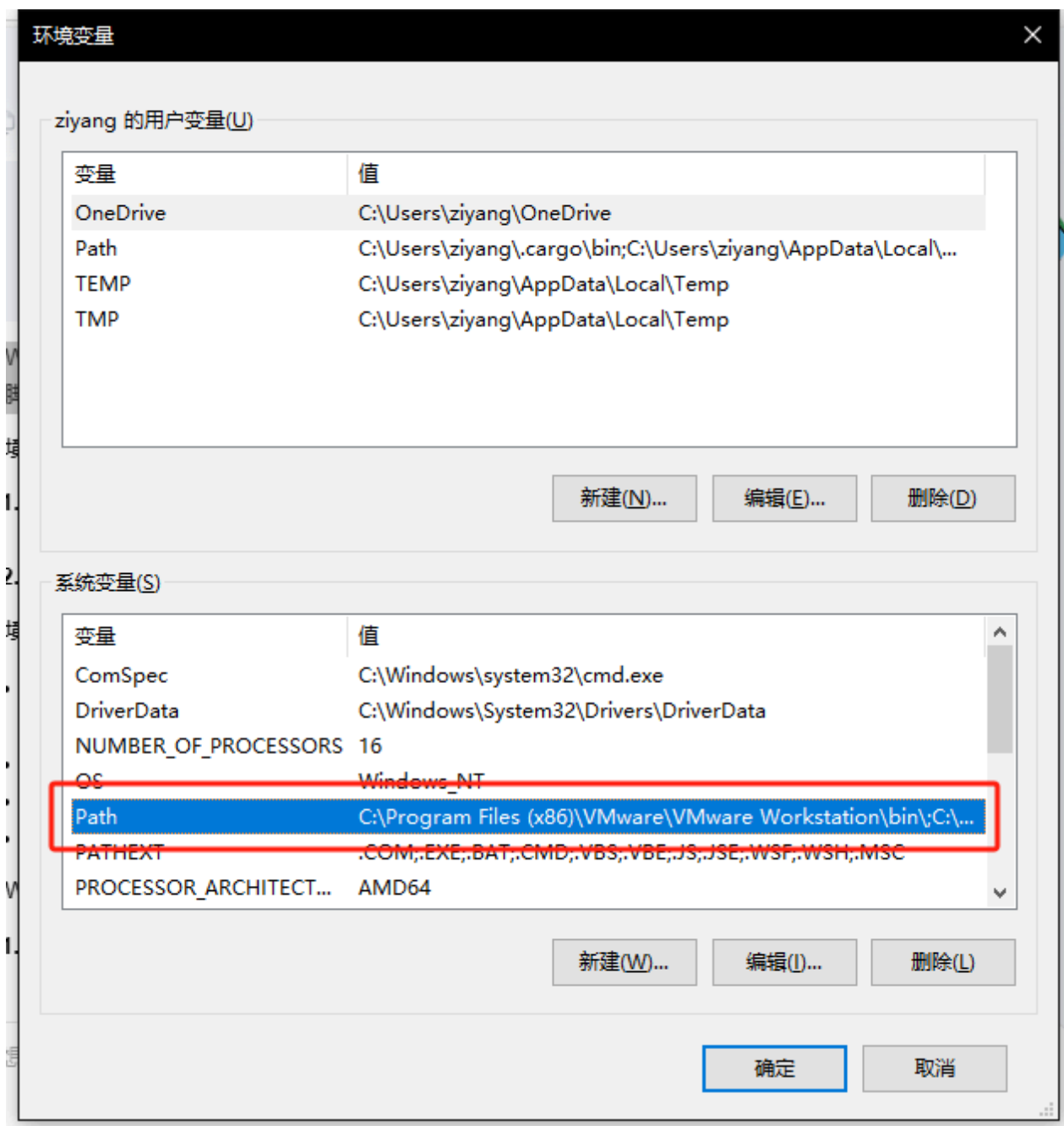
在搜索框中搜索【环境变量】，



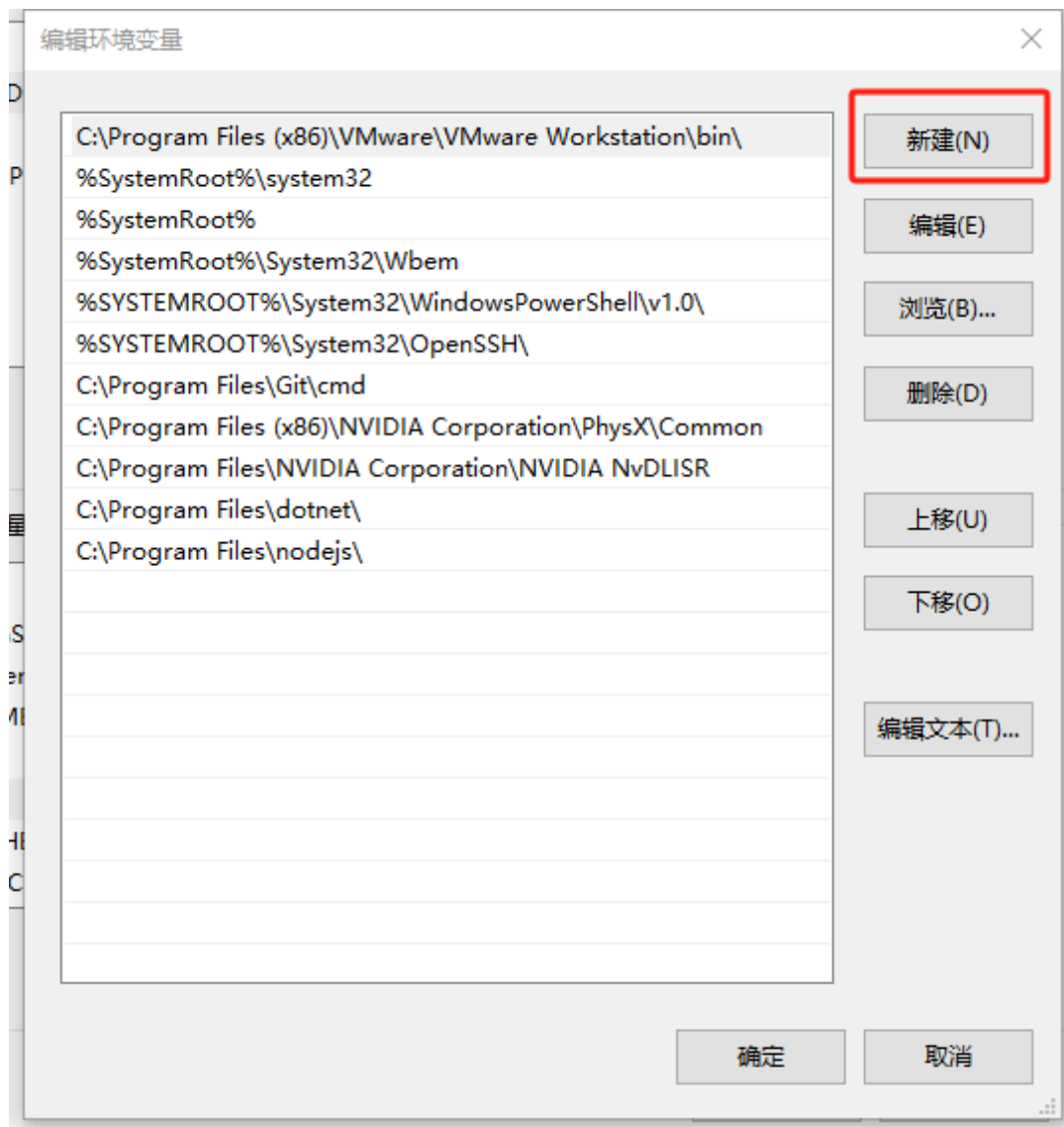
之后点击【编辑系统环境变量】，弹出以下对话框：



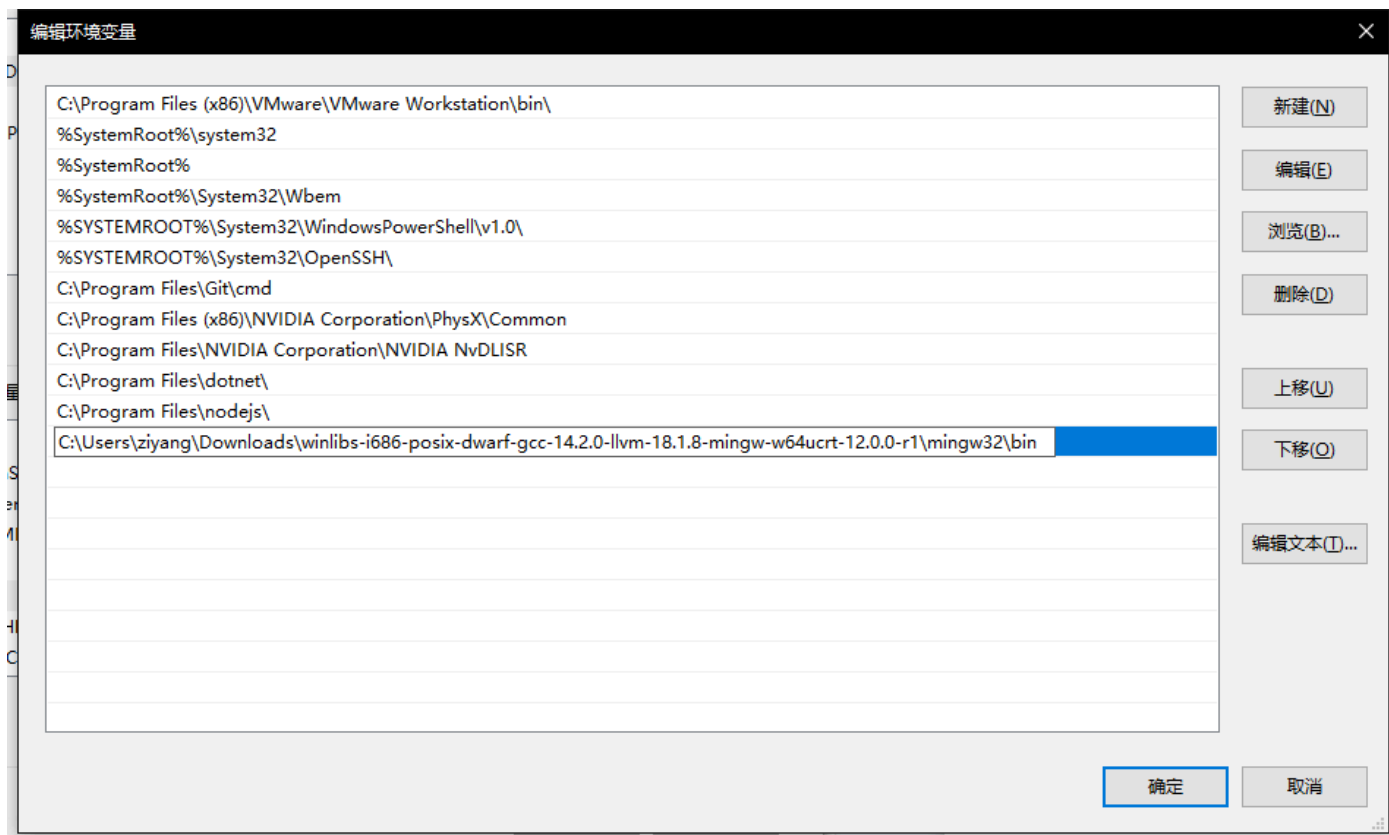
点击【环境变量】，弹出以下对话框



选中红框中的Path，点击【编辑】，弹出以下对话框



点击【新建】按钮，



获取你解压后的mingw文件夹中的mingw32/bin的路径，例如，我解压出的mingw路径是
C:\Users\ziyang\Downloads\winlibs-i686-posix-dwarf-gcc-14.2.0-llvm-18.1.8-mingw-w64ucrt-12.0.0-r1，我就在该路径后面填上\mingw32\bin，变成
C:\Users\ziyang\Downloads\winlibs-i686-posix-dwarf-gcc-14.2.0-llvm-18.1.8-mingw-w64ucrt-12.0.0-r1\mingw32\bin

写好之后一直点【确定】按钮即可

此时**重新开启一个CMD**，向其中输入gcc --version，如果出现下图所示情况，则说明MinGW配置成功！

```
命令提示符
Microsoft Windows [版本 10.0.19045.4894]
(c) Microsoft Corporation。保留所有权利。

C:\Users\ziyang>gcc --version
gcc (MinGW-W64 i686-ucrt-posix-dwarf, built by Brecht Sanders, r1) 14.2.0
Copyright (C) 2024 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.

C:\Users\ziyang>
```

Step2: 配置VSCode

VSCode最大的一个特点就是能够兼容大多数编程语言，我们只需要安装相应的扩展便可完成对某些语言的支持。以往的各种IDE（集成开发环境）方向都比较专一，能支持的变成语言相对较少。例如Devcpp，CodeBlocks，CLion等主要支持C/C++；Idea，Eclipse等主要支持Java；Pycharm，Spider等主要支持Python

下载VSCode

官网地址（可能会有点慢）：<https://code.visualstudio.com/>

镜像地址（需要连接校园网）：<http://202.112.113.225/VSCodeUserSetup-x64-1.91.1.exe>

安装VSCode



选择目标位置

您想将 Visual Studio Code 安装在什么地方？



安装程序将安装 Visual Studio Code 到下列文件夹中。

单击“下一步”继续。如果您想选择其它文件夹，单击“浏览”。

C:\Users\ziyang\AppData\Local\Programs\Microsoft VS Code

浏览(R)...

至少需要有 377.4 MB 的可用磁盘空间。

< 上一步(B)

下一步(N) >

取消

选择开始菜单文件夹

您想在哪里放置程序的快捷方式？



安装程序现在将在下列开始菜单文件夹中创建程序的快捷方式。

单击“下一步”继续。如果您想选择其它文件夹，单击“浏览”。

Visual Studio Code

浏览(R)...

☐ 不创建开始菜单文件夹(D)

< 上一步(B)

下一步(N) >

取消

选择附加任务

您想要安装程序执行哪些附加任务？



选择您想要安装程序在安装 Visual Studio Code 时执行的附加任务，然后单击“下一步”。

附加快捷方式：

☒ 创建桌面快捷方式(D)

其他：

☐ 将“通过 Code 打开”操作添加到 Windows 资源管理器文件上下文菜单

☐ 将“通过 Code 打开”操作添加到 Windows 资源管理器目录上下文菜单

☒ 将 Code 注册为受支持的文件类型的编辑器

☒ 添加到 PATH (重启后生效)

< 上一步(B)

下一步(N) >

取消

准备安装

安装程序现在准备开始安装 Visual Studio Code 到您的电脑中。



单击“安装”继续此安装程序。如果您想要回顾或改变设置，请单击“上一步”。

目标位置:

C:\Users\ziyang\AppData\Local\Programs\Microsoft VS Code

开始菜单文件夹:

Visual Studio Code

附加任务:

附加快捷方式:

创建桌面快捷方式(D)

其他:

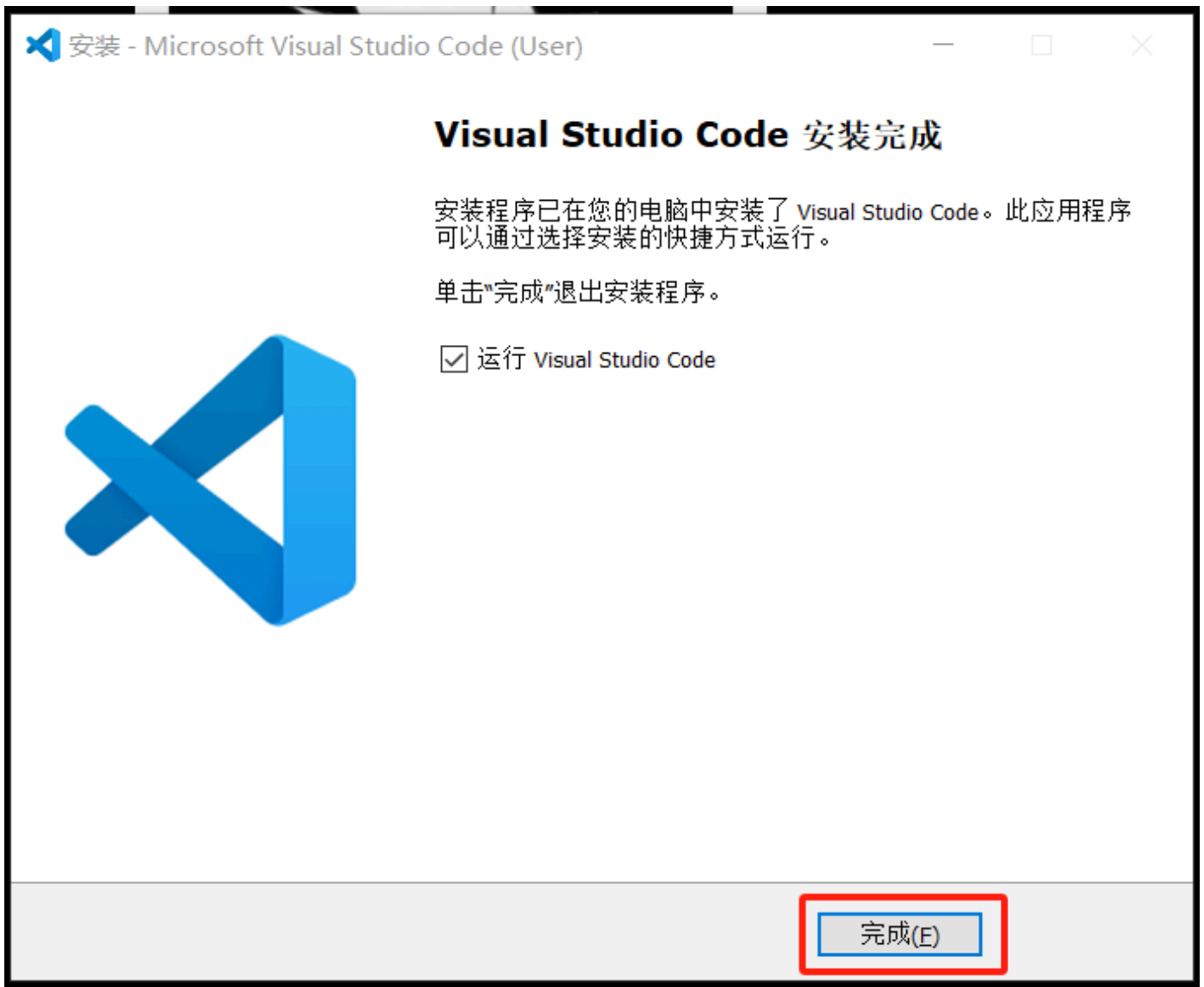
将 Code 注册为受支持的文件类型的编辑器

添加到 PATH (重启后生效)

< 上一步(B)

安装(I)

取消



配置C/C++运行环境

下载安装扩展

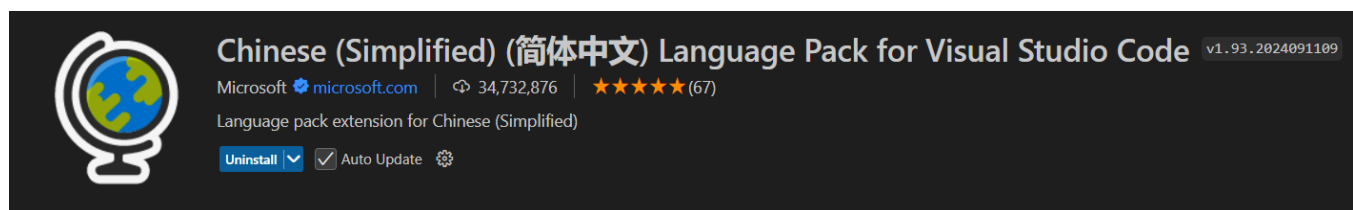
进入VSCode后点击左边菜单栏的扩展（Extensions），在扩展市场中搜索相应插件并安装



需要安装以下3款插件：

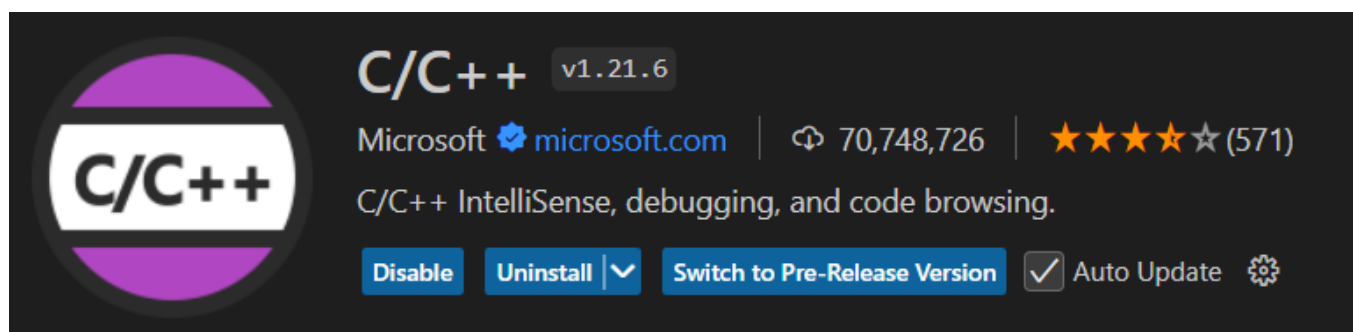
- Chinese (Simplified) (简体中文)

VSCode界面默认是英文的，可以选择安装中文翻译



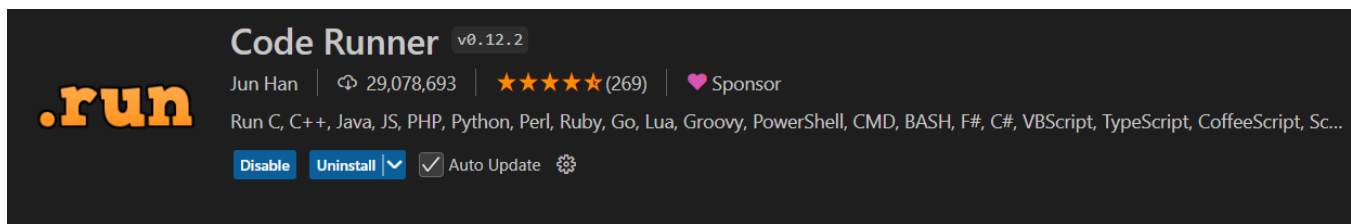
- C/C++

提供C/C++语言的支持，包括代码补全，调试等



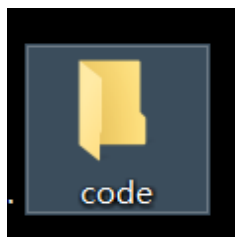
- Code Runner

能够运行C，C++等多种代码的扩展

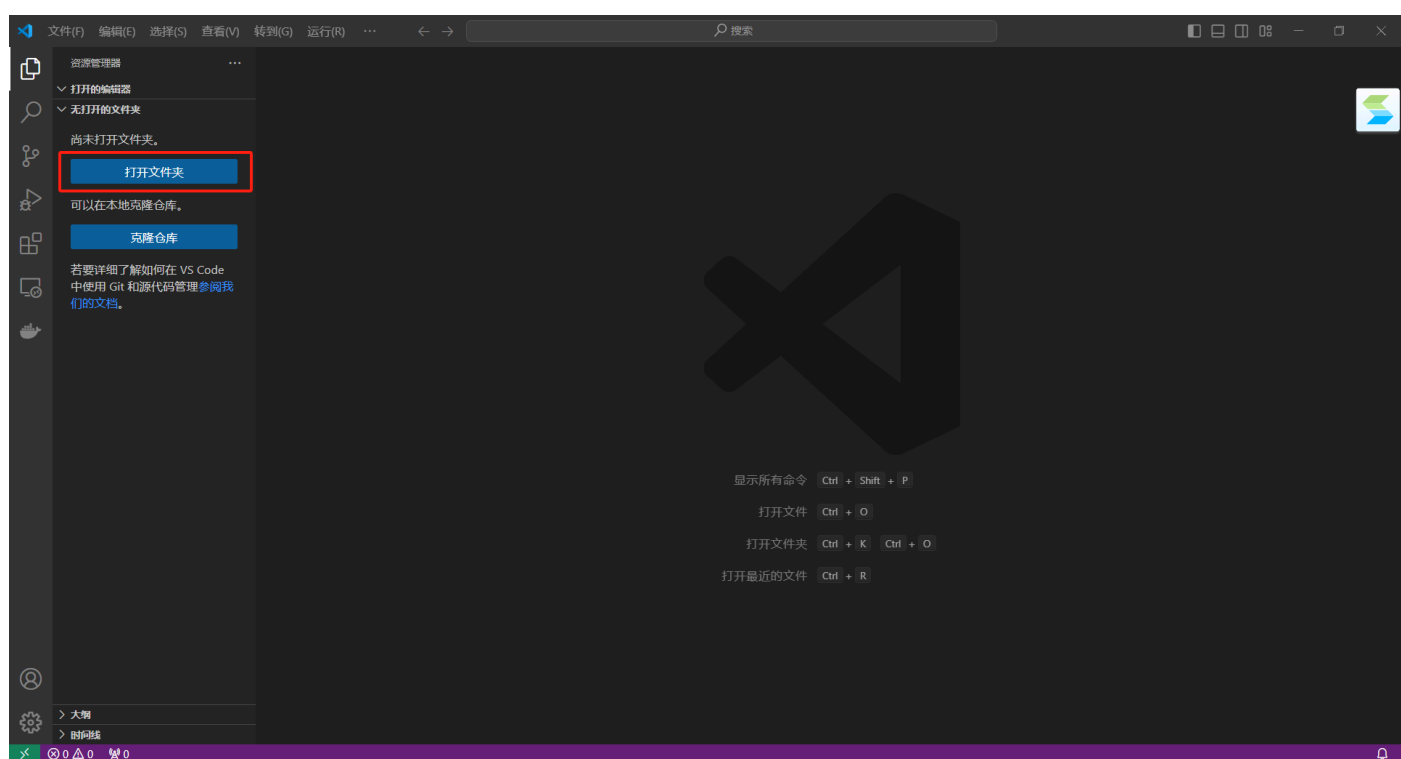


新建并打开目录

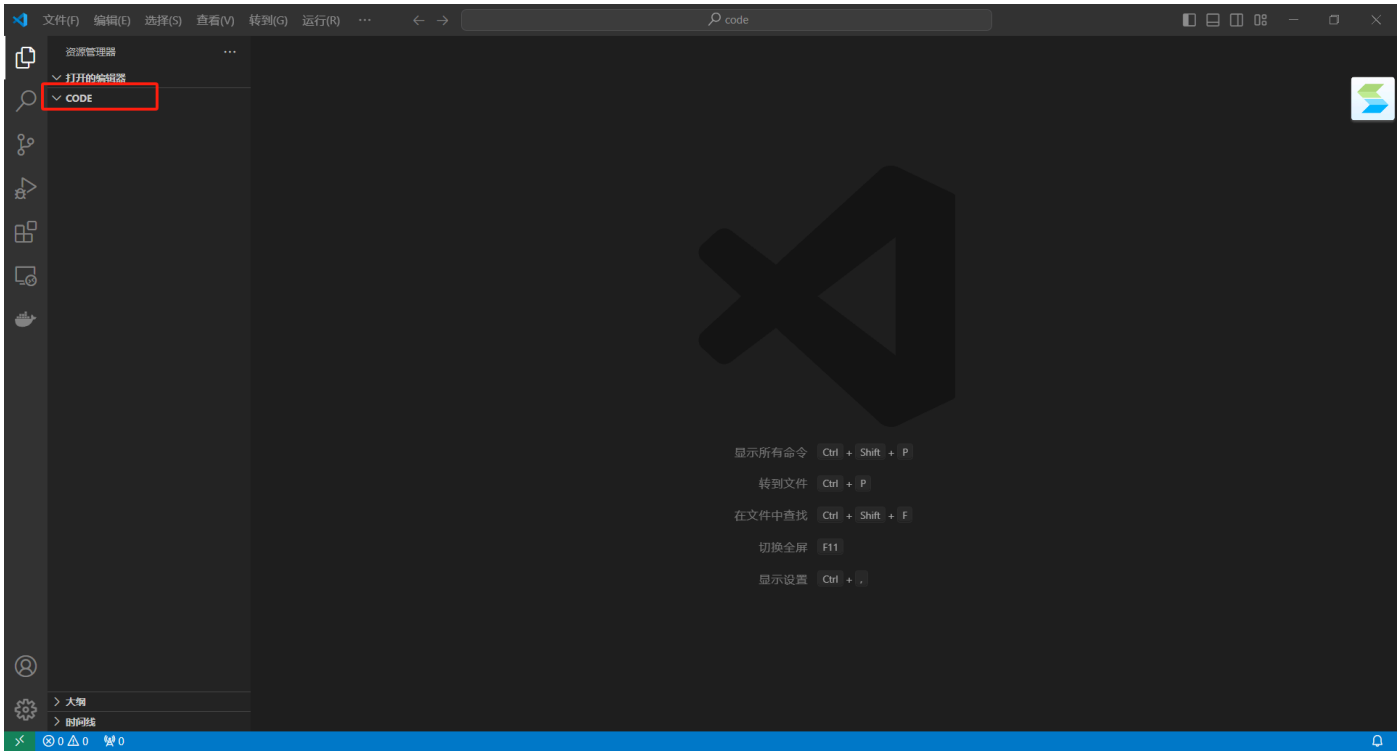
我们需要新建一个目录来存放代码，在这里我在桌面上新建了一个目录



点击【打开文件夹】，选中目标文件夹并打开

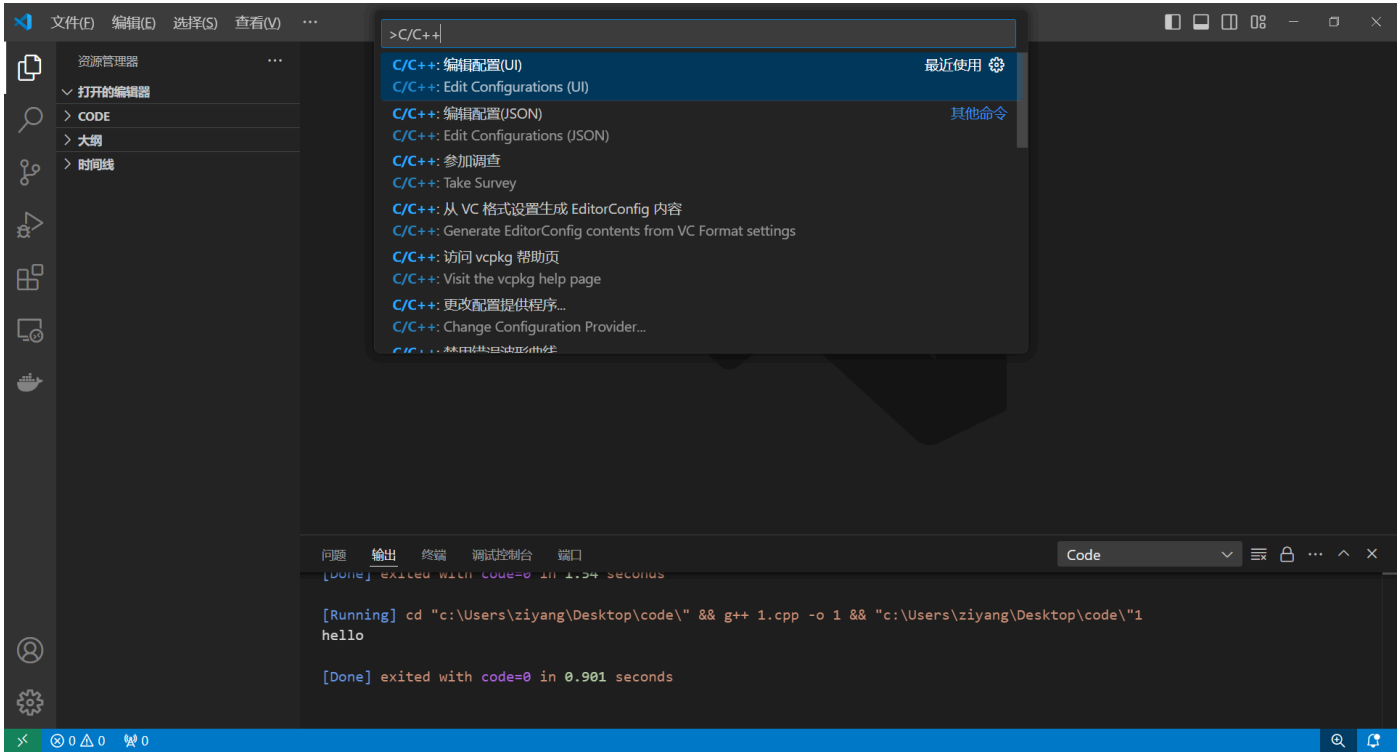


如下图所示，当红框中内容变成文件夹名称，则说明已经打开对应目录

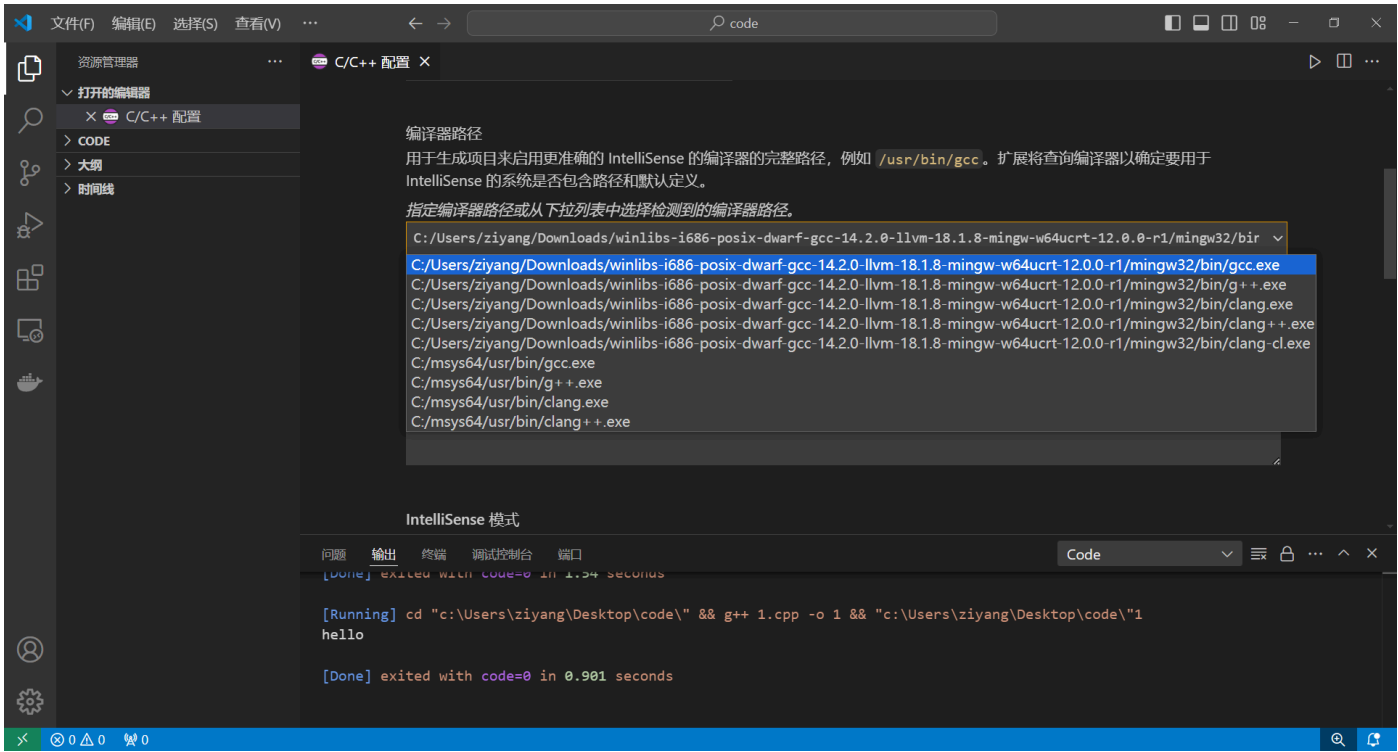


配置C/C++扩展

按下**Ctrl+Shift+P**，输入C/C++，选择【C/C++：编辑配置（UI）】

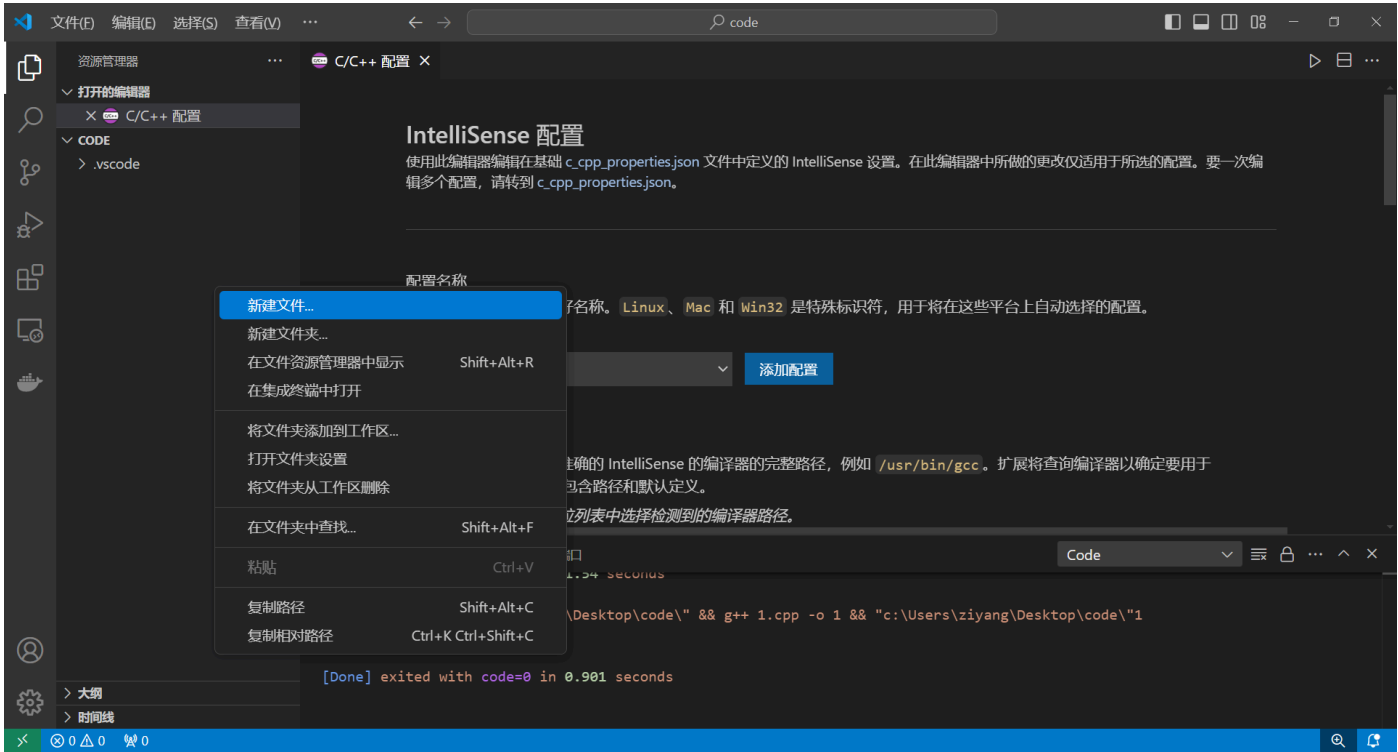


找到编译器路径，选择Step1中，你存放MinGW的路径，并选中其中的gcc.exe，如下图所示：
选择完成之后关闭该配置页面即可。

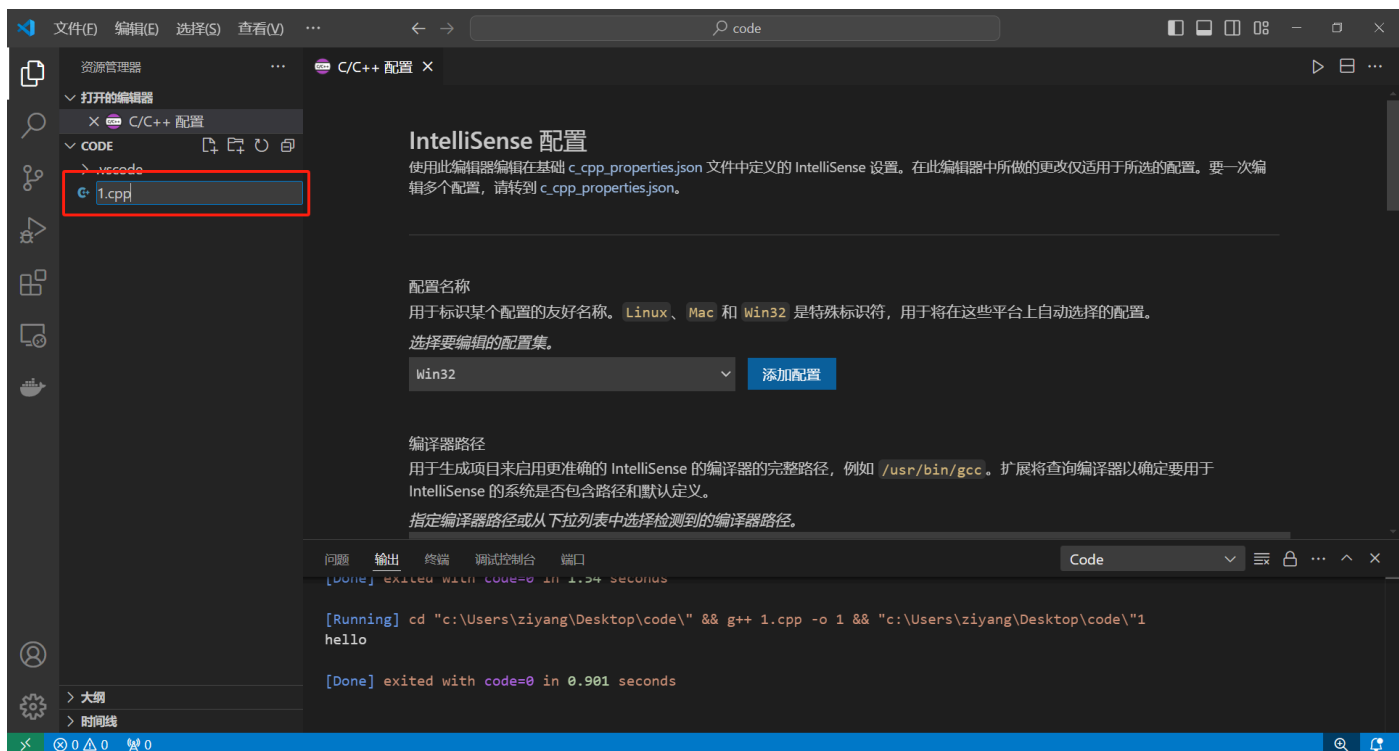


新建文件并运行

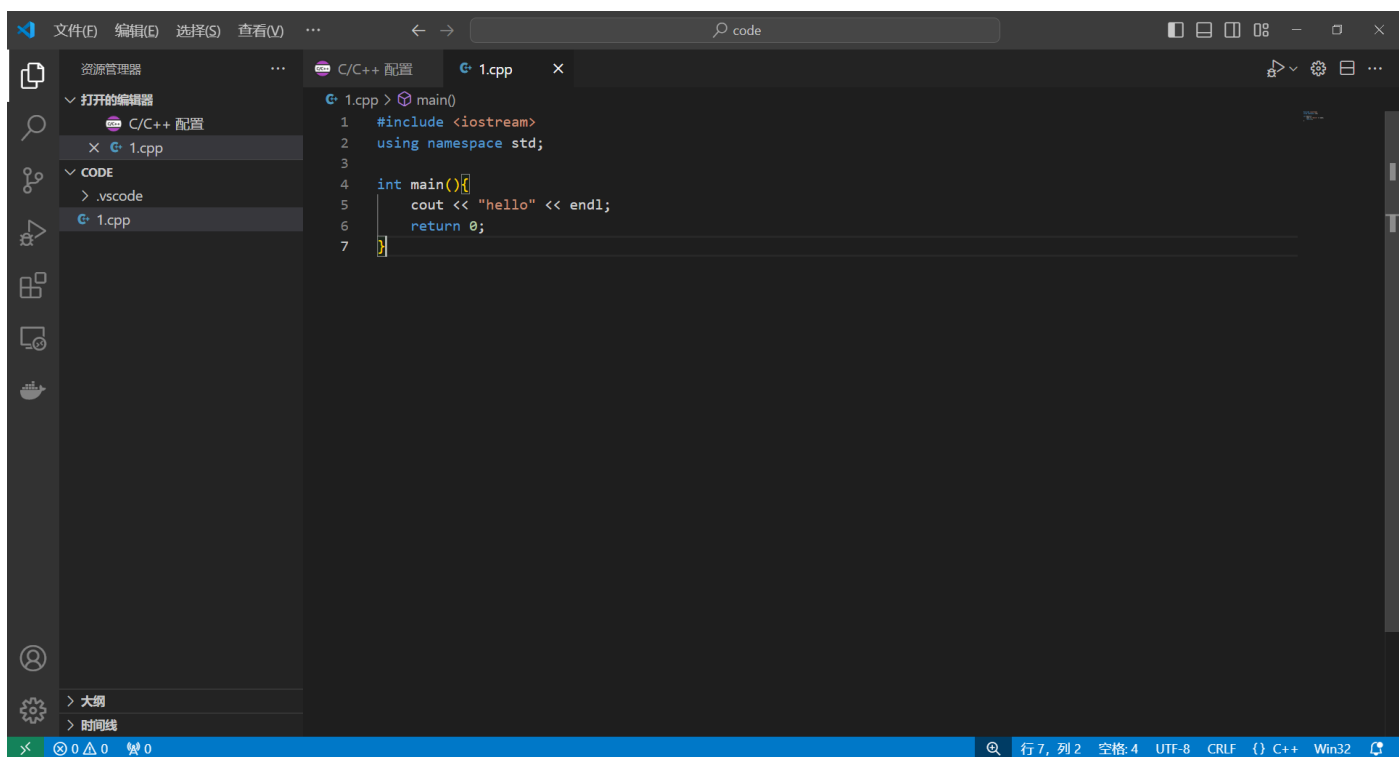
在空白处右击，选择【新建文件】



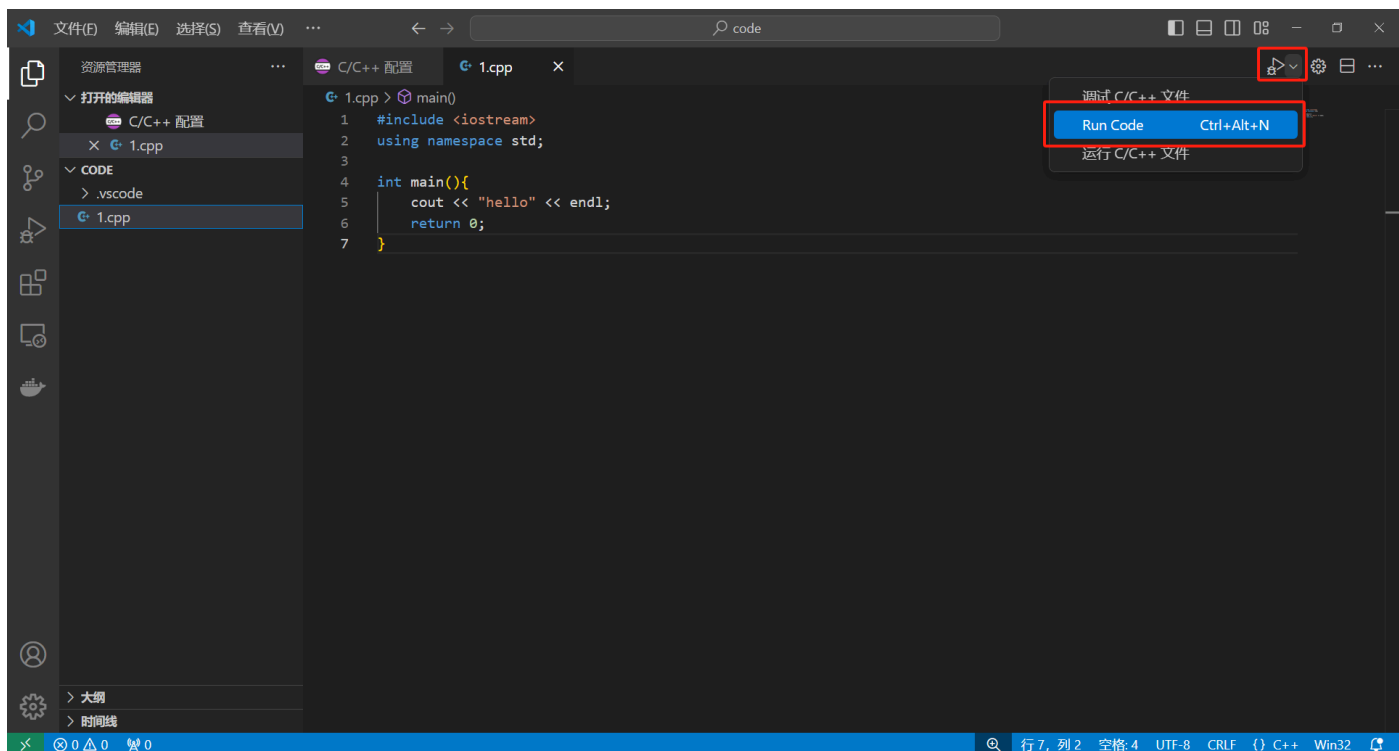
输入新建文件名，如图所示，我们以1.cpp为例



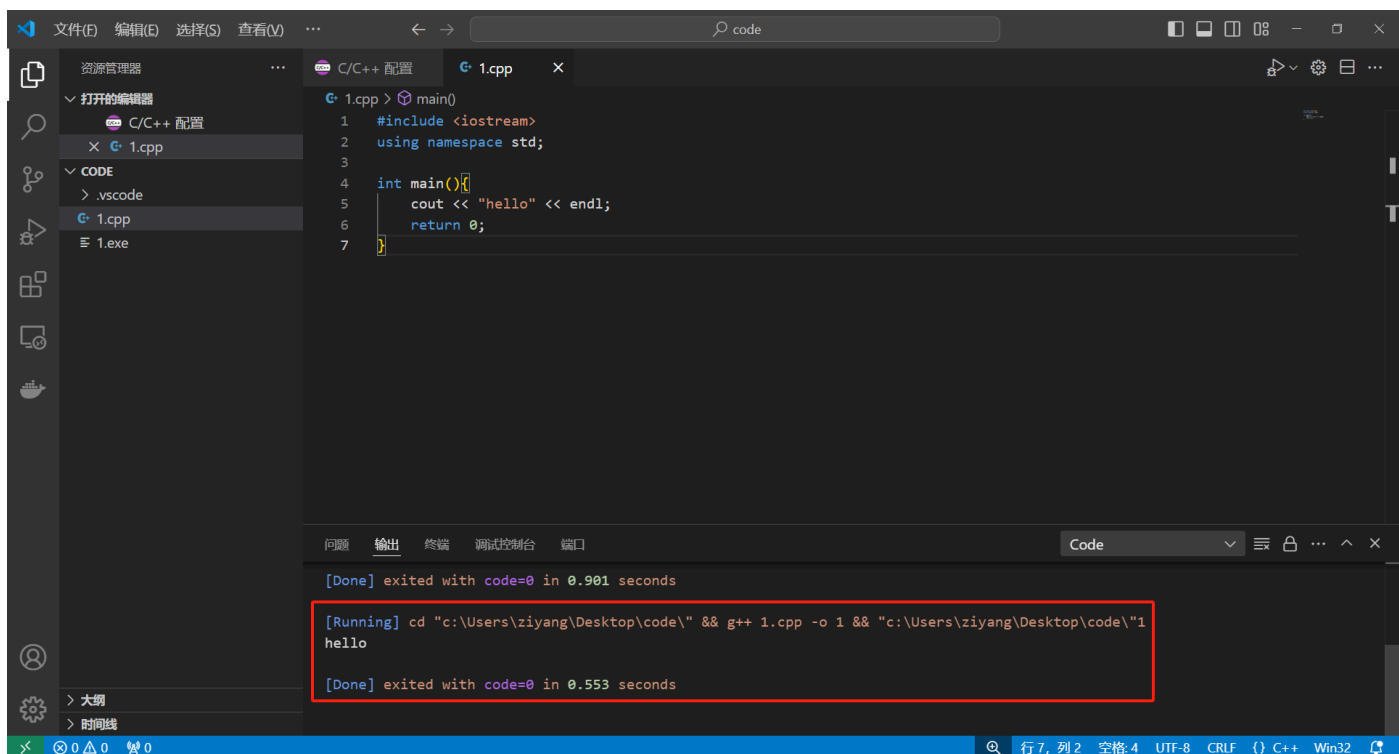
编写cpp程序，写完之后需要按下Ctrl+S保存，**每次写完程序之后都要记得保存，否则你对文件的修改将不起作用**



点击右上角运行旁边的下拉按钮，在下拉框中选择【Run Code】

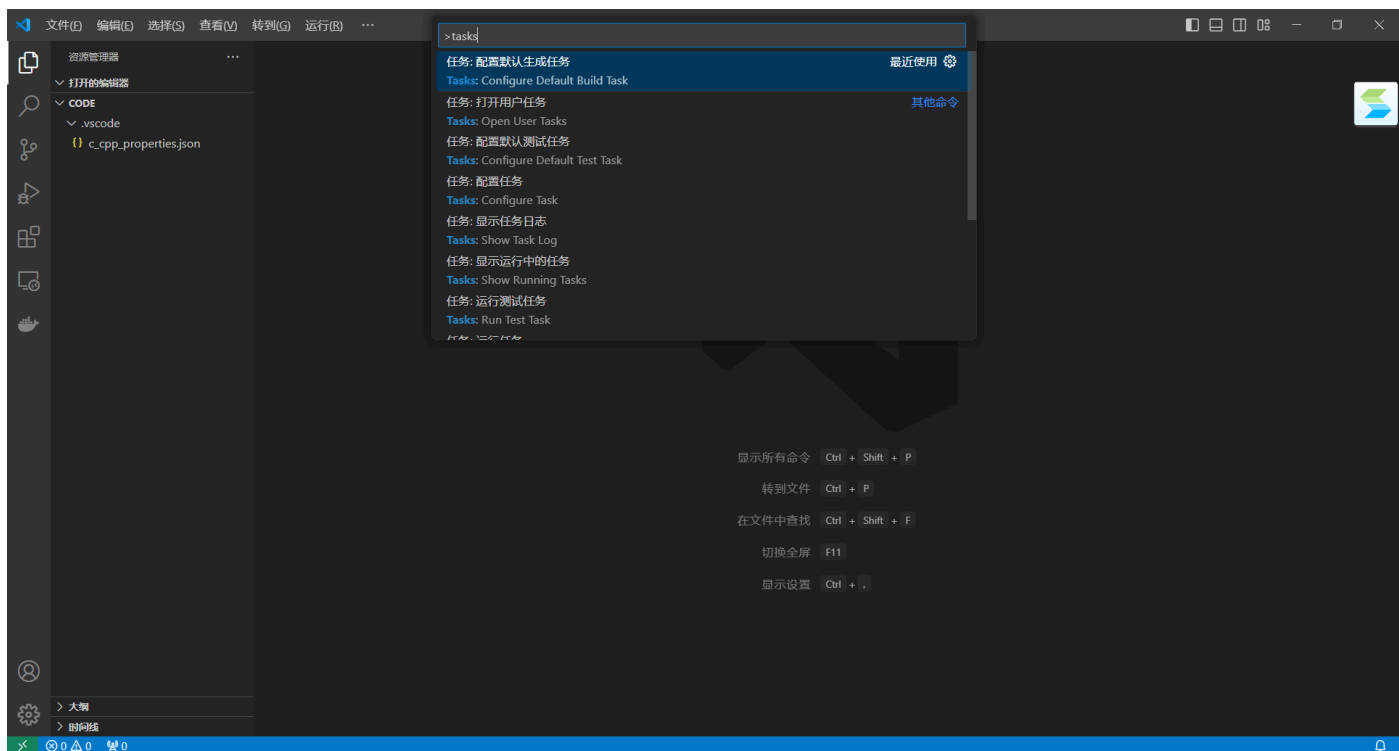


查看结果，运行之后可以在下方的窗口中看到程序的输出，这表明我们配置成功！

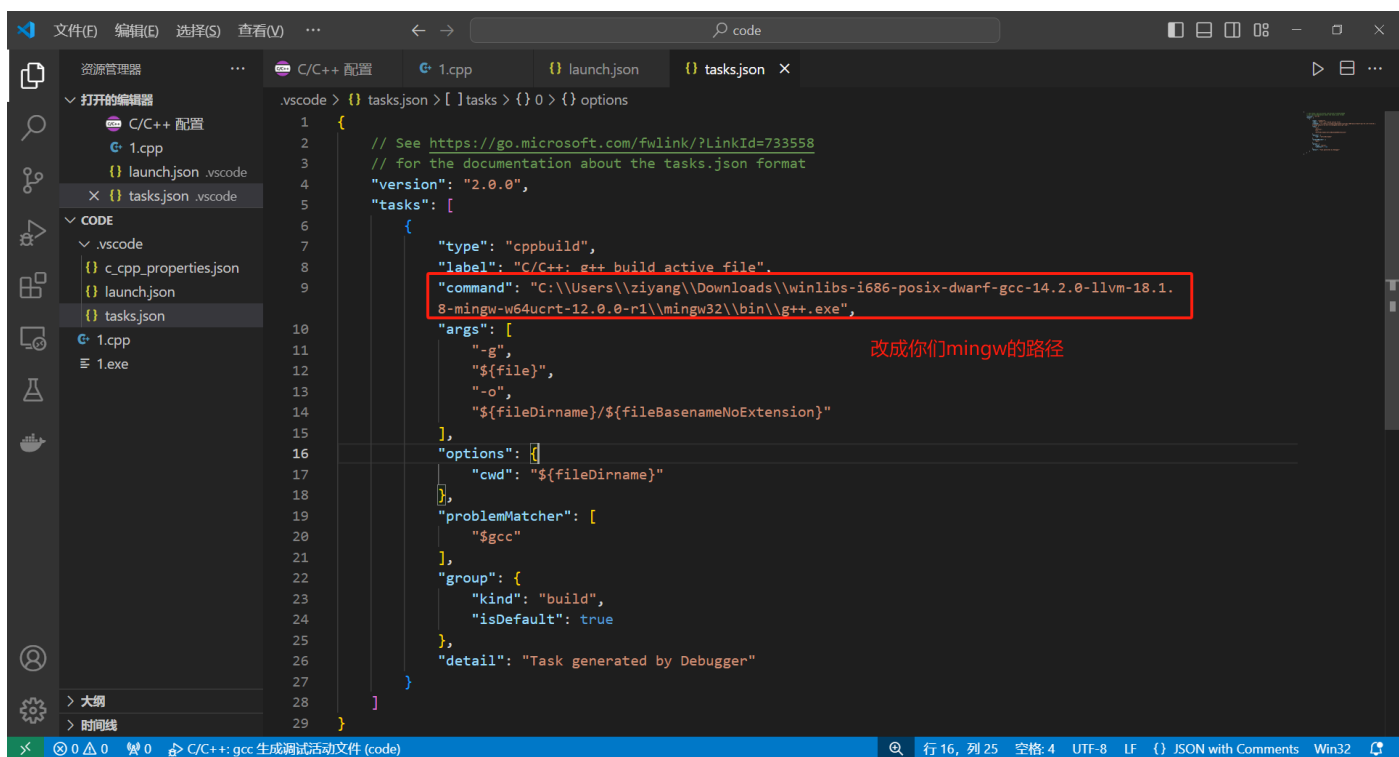


配置C/C++调试环境

按下**Ctrl+Shift+P**，输入tasks，选择【任务：配置默认生成任务】



点击之后VSCode会在.vscode目录下新建一个tasks.json，我们需要将tasks.json填写成以下这样：



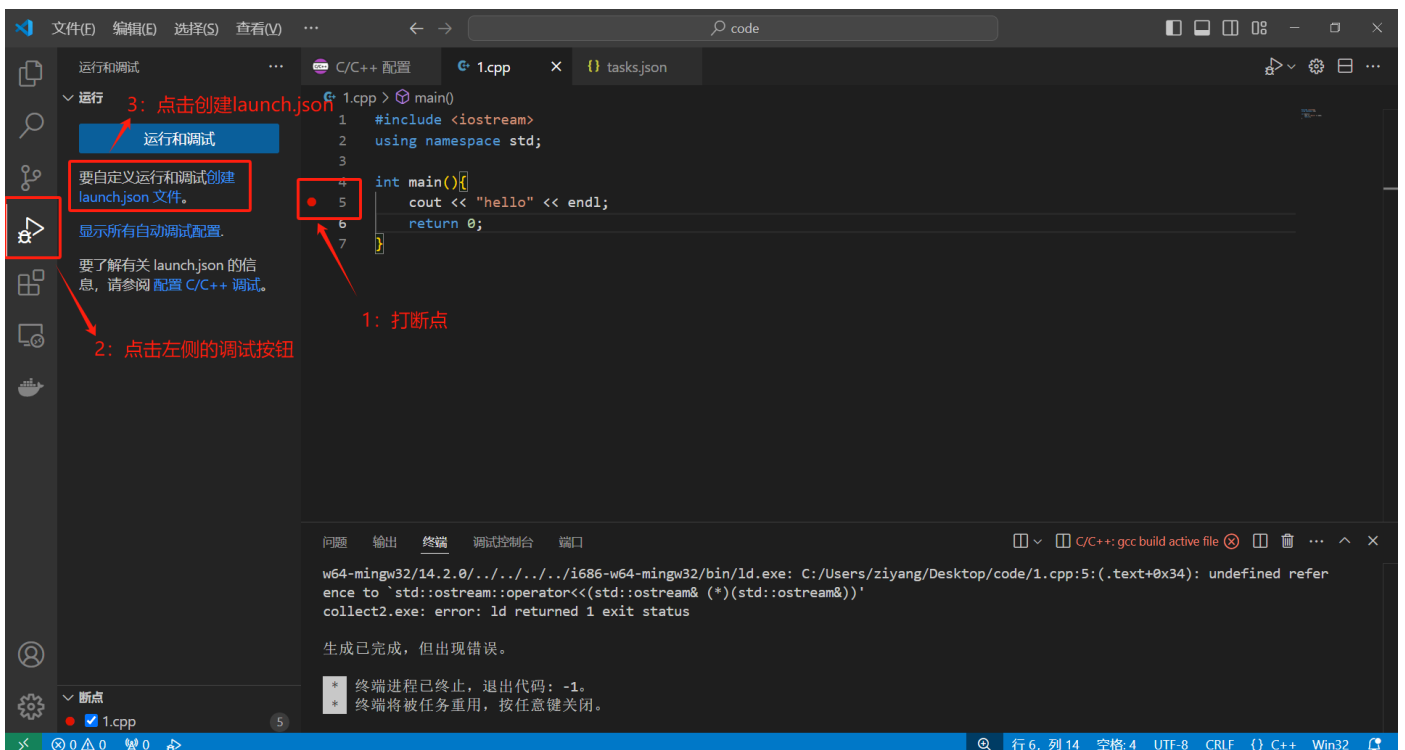
```
1 {
2     // See https://go.microsoft.com/fwlink/?LinkId=733558
3     // for the documentation about the tasks.json format
4     "version": "2.0.0",
5     "tasks": [
6         {
```

```

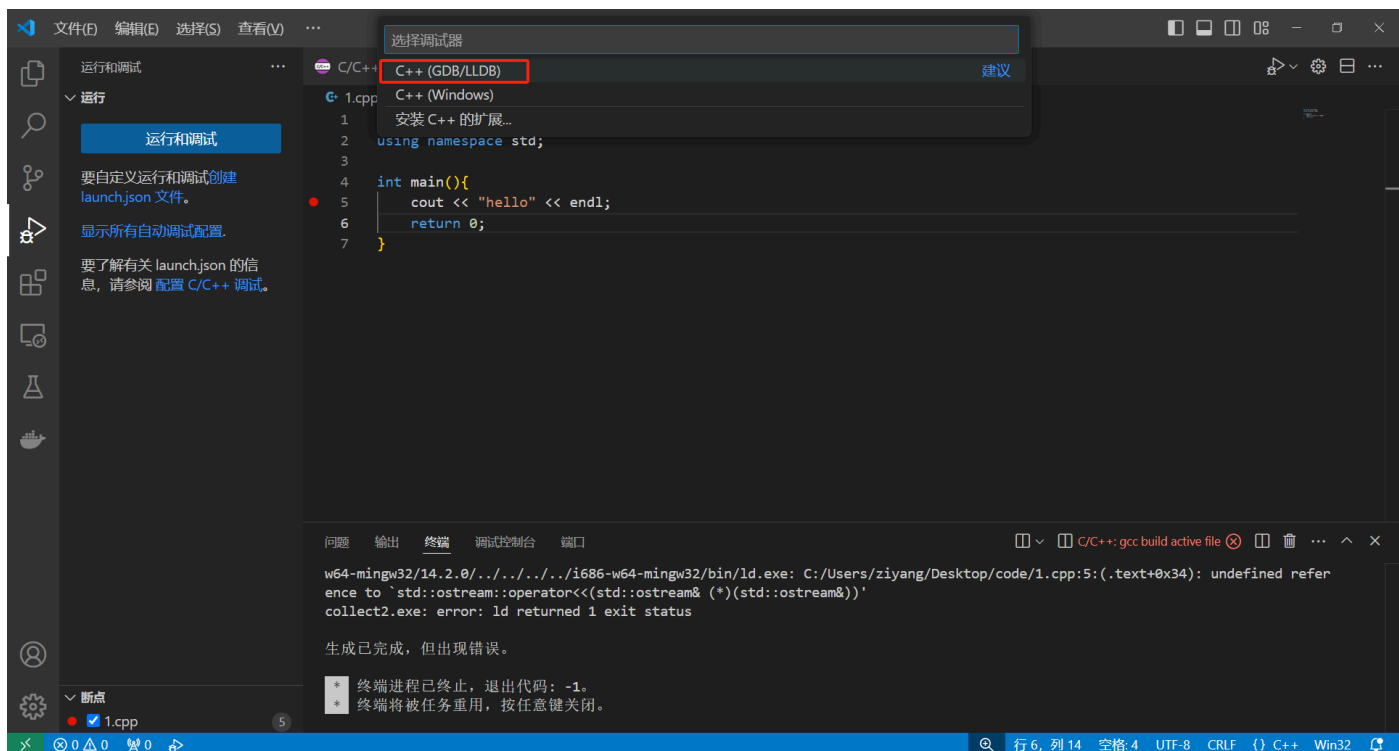
7      "type": "cppbuild",
8      "label": "C/C++: g++ build active file",
9      "command": "C:\\Users\\ziyang\\Downloads\\winlibs-i686-posix-
dwarf-gcc-14.2.0-llvm-18.1.8-mingw-w64ucrt-12.0.0-r1\\mingw32\\bin\\g++.exe",
10     "args": [
11         "-g",
12         "${file}",
13         "-o",
14         "${fileDirname}/${fileBasenameNoExtension}"
15     ],
16     "options": {
17         "cwd": "${fileDirname}"
18     },
19     "problemMatcher": [
20         "$gcc"
21     ],
22     "group": {
23         "kind": "build",
24         "isDefault": true
25     },
26     "detail": "Task generated by Debugger"
27 }
28 ]
29 }

```

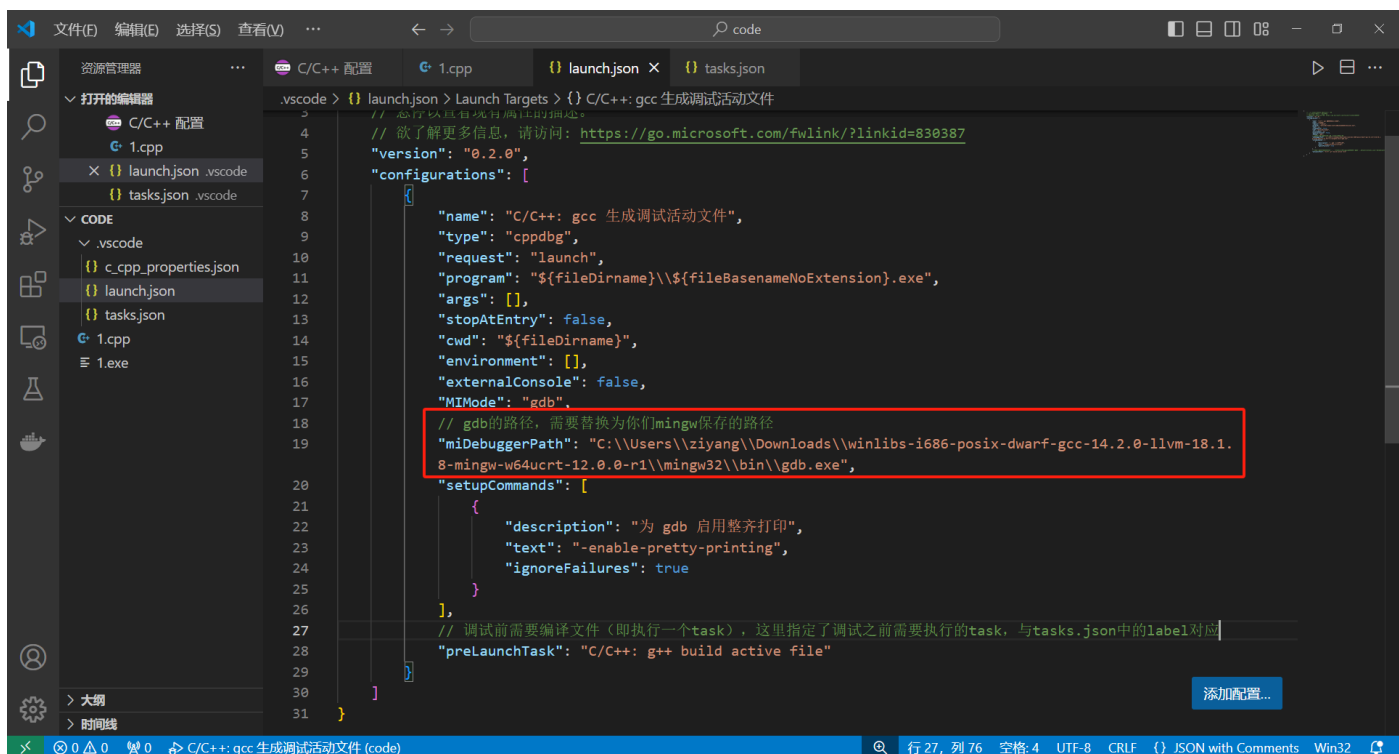
在打好断点之后，点击左侧的调试按钮，选择创建launch.json



选择【C++ (GDB/LLDB)】



编辑launch.json为如下，需要注意，红框中应该改成你们MinGW保存的路径



```
1 {
2     // 使用 IntelliSense 了解相关属性。
3     // 悬停以查看现有属性的描述。
4     // 欲了解更多信息，请访问: https://go.microsoft.com/fwlink/?linkid=830387
5     "version": "0.2.0",
6     "configurations": [
7         {
8             "name": "C/C++: gcc 生成调试活动文件",
```

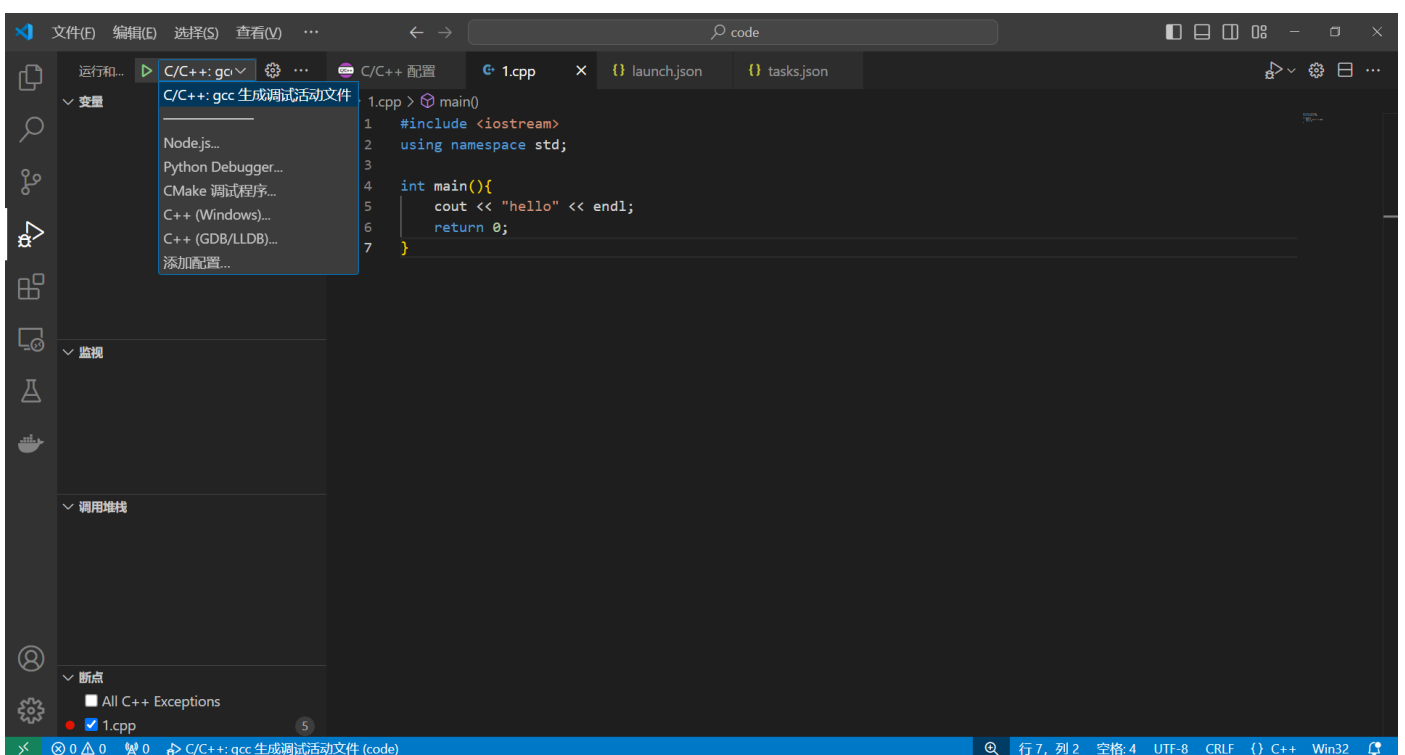


```

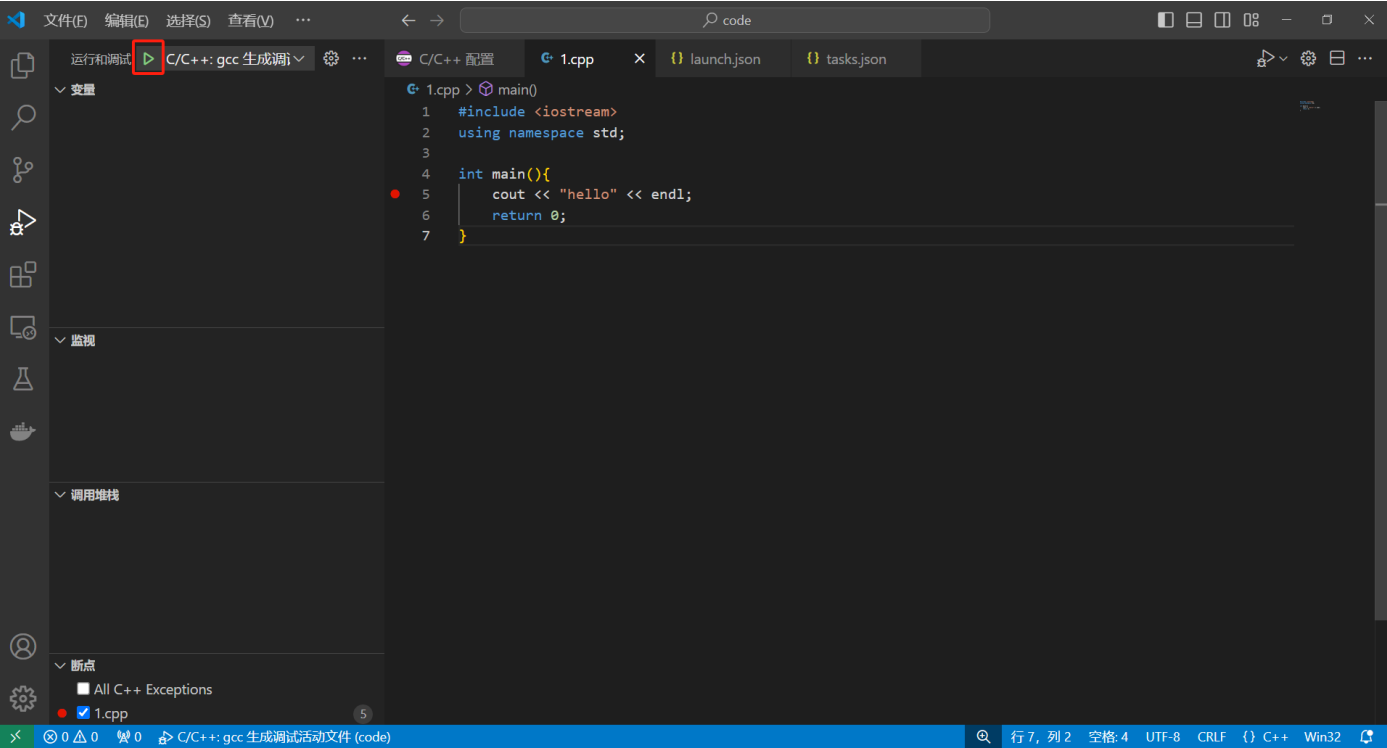
9      "type": "cppdbg",
10     "request": "launch",
11     "program": "${fileDirname}\\${fileBasenameNoExtension}.exe",
12     "args": [],
13     "stopAtEntry": false,
14     "cwd": "${fileDirname}",
15     "environment": [],
16     "externalConsole": false,
17     "MIMode": "gdb",
18     // gdb的路径，需要替换为你们mingw保存的路径
19     "miDebuggerPath": "C:\\\\Users\\\\ziyang\\\\Downloads\\\\winlibs-i686-
posix-dwarf-gcc-14.2.0-llvm-18.1.8-mingw-w64ucrt-12.0.0-
r1\\\\mingw32\\\\bin\\\\gdb.exe",
20     "setupCommands": [
21     {
22         "description": "为 gdb 启用整齐打印",
23         "text": "-enable-pretty-printing",
24         "ignoreFailures": true
25     }
26 ],
27     // 调试前需要编译文件（即执行一个task），这里指定了调试之前需要执行的task，与
tasks.json中的label对应
28     "preLaunchTask": "C/C++: g++ build active file"
29 }
30 ]
31 }

```

回到调试窗口，选择我们刚刚配置的【C/C++: gcc 生成调试活动文件】



点击绿色的按钮便可开始调试



可以看到此时程序停到我们打断点处，说明C/C++调试环境配置成功

