vscode配置C/C++环境

文章目录

准备工作

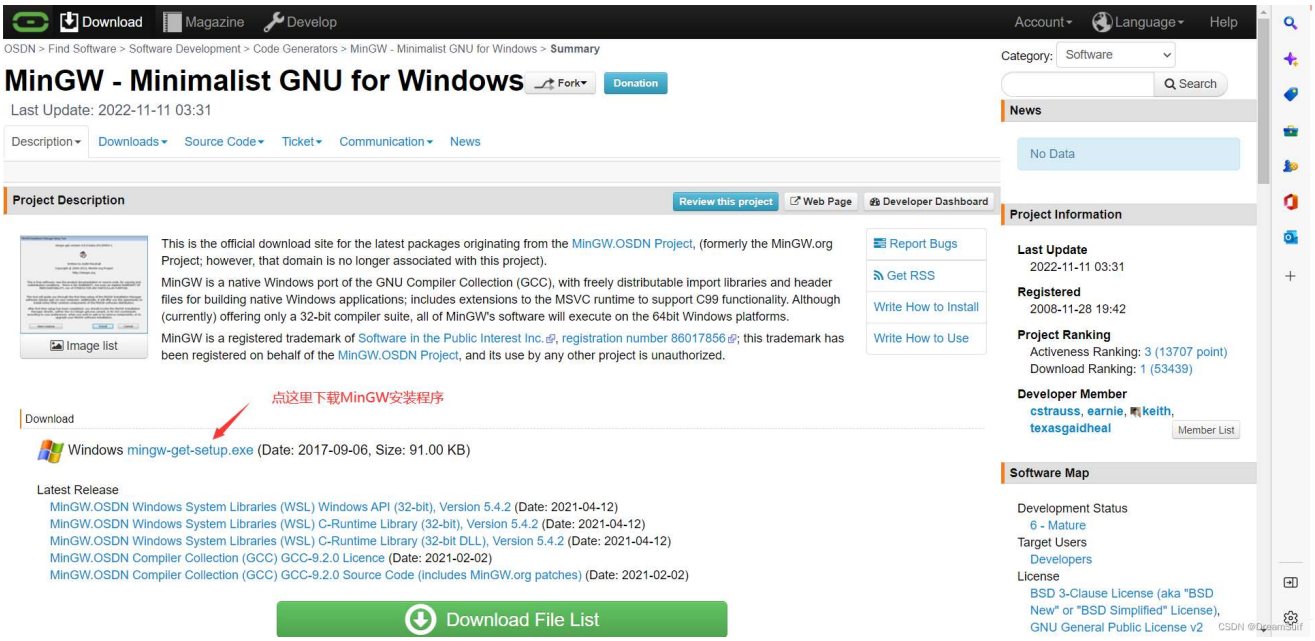
在 Vscode 里配置C/C++运行环境，首先，需要下载 C/C++ 的开发环境，然后将 C/C++ 的开发环境添加至系统变量中。

第一步 下载vscode

> VsCode大家应该都会下，这里就不提供VsCode的下载与安装教程了

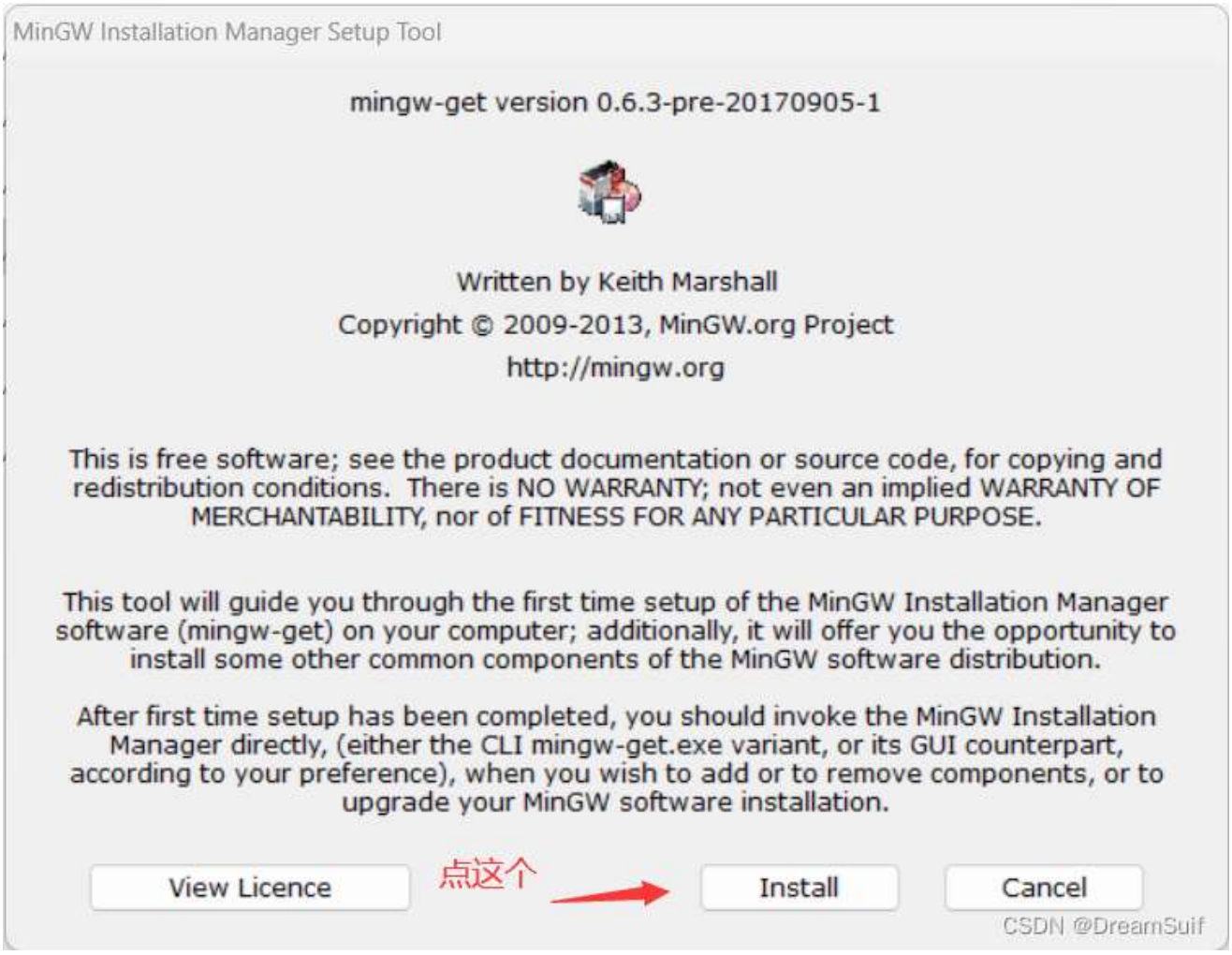
第二步 下载mingw

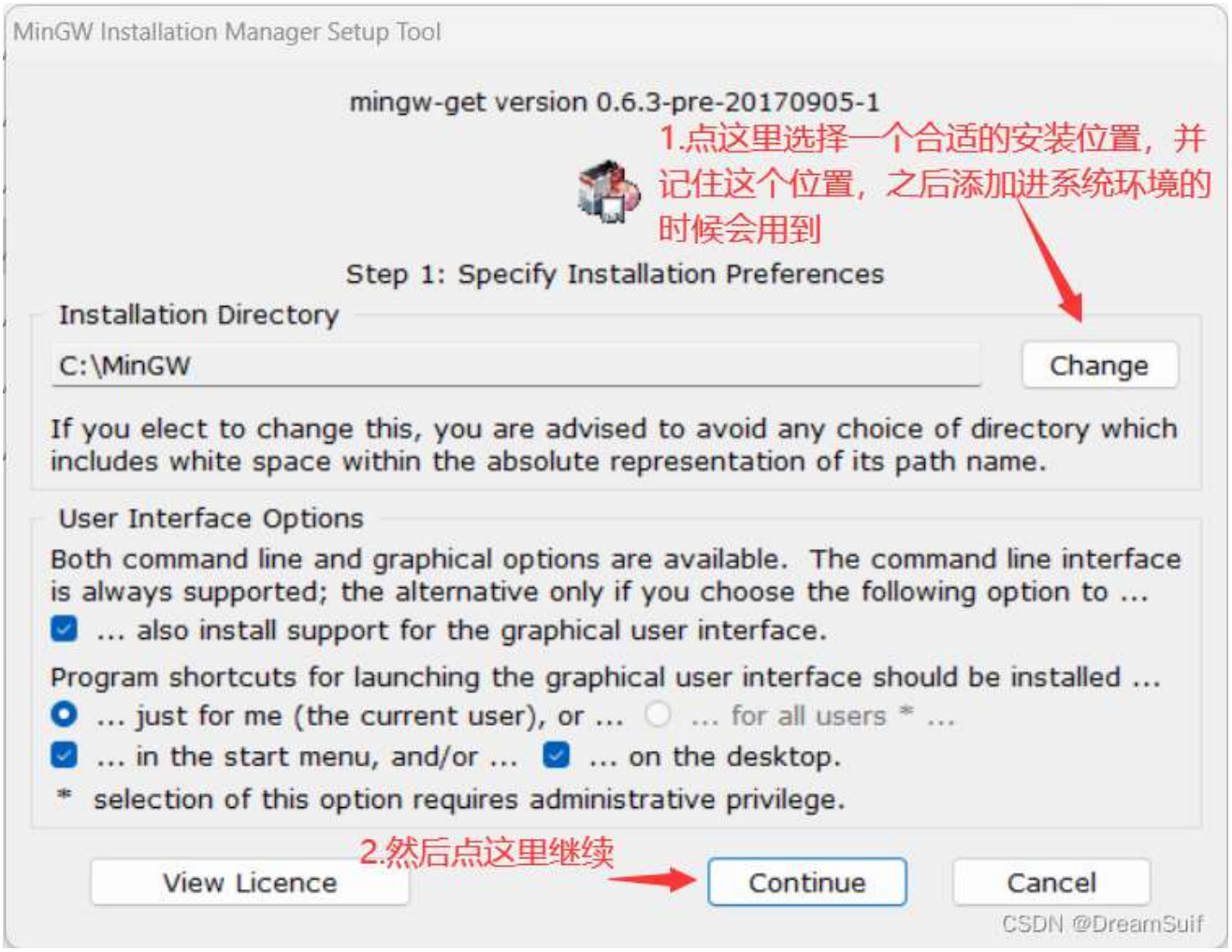
这里采用mingw作为 C/C++ 开发环境，官网链接如下
官网链接：MinGW官网

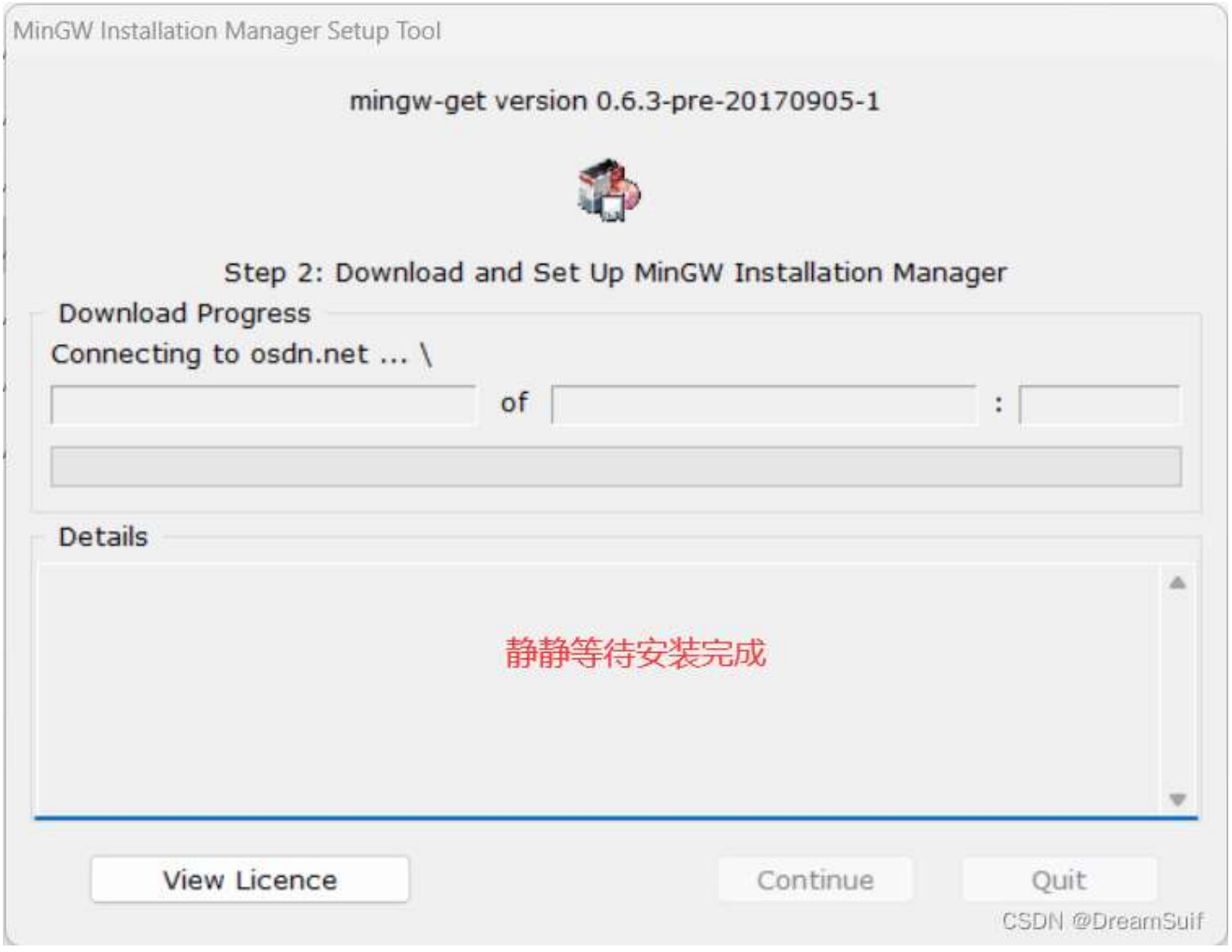


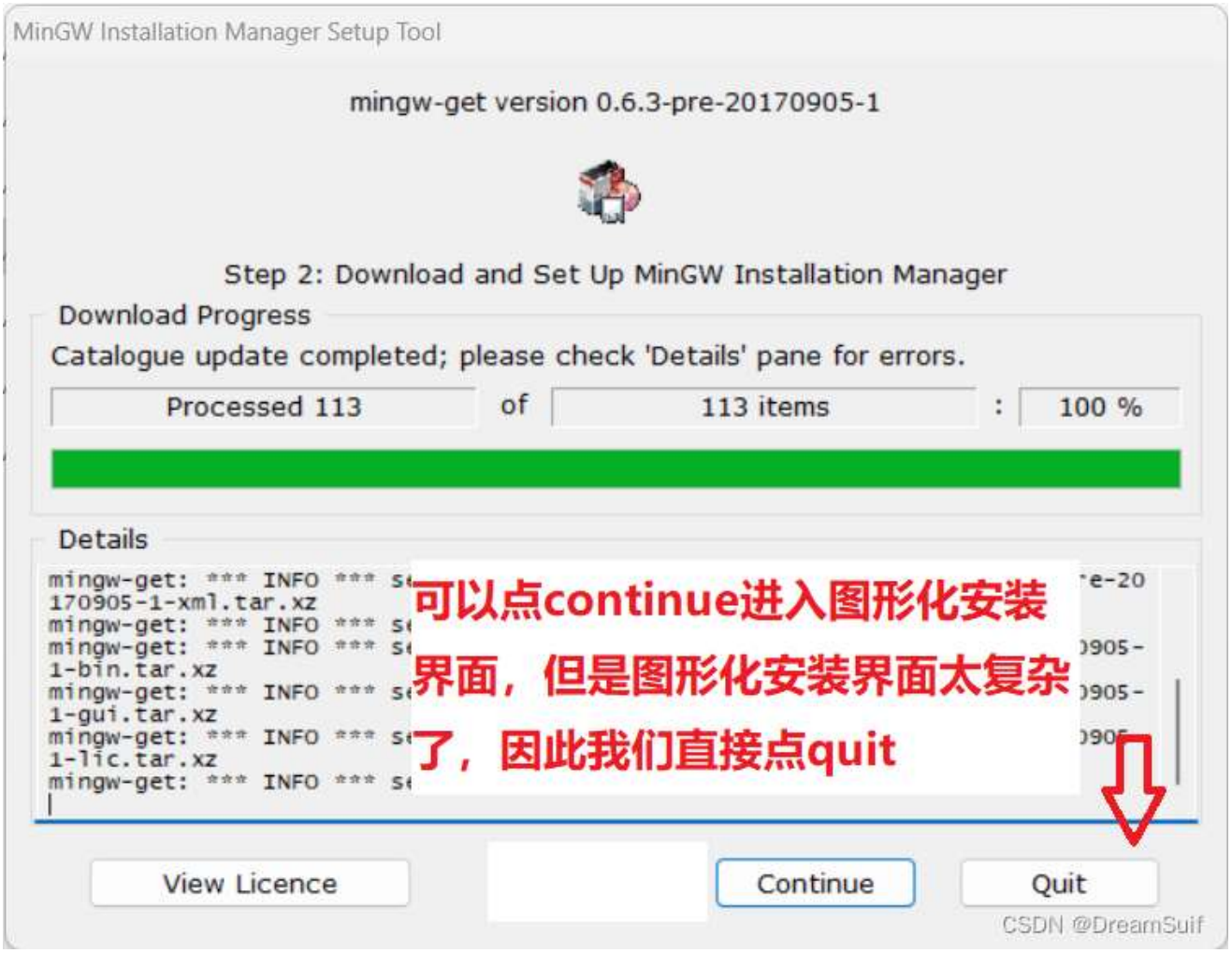
> 下载完成后我们会得到这样一个安装程序

> 双击打开

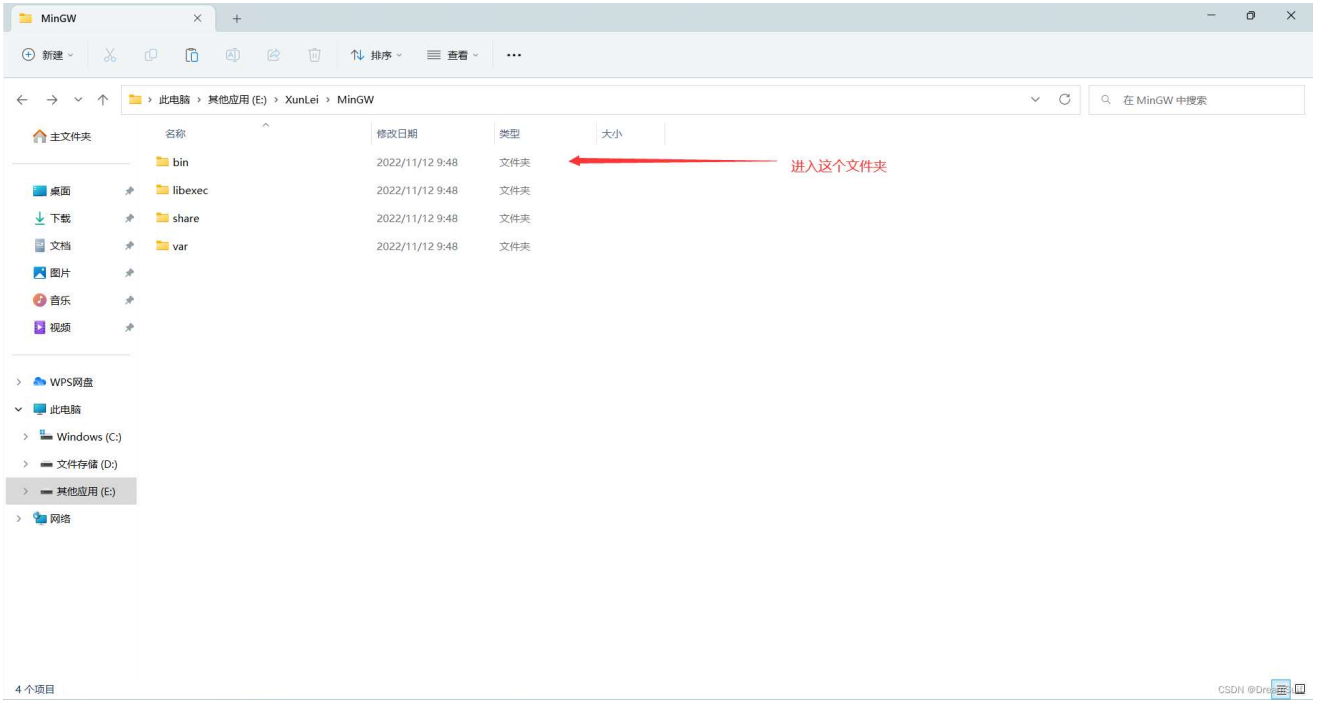

MinGW Installation Manager Setup Tool
mingw-get version 0.6.3-pre-20170905-1
Written by Keith Marshall
Copyright © 2009-2013, MinGW.org Project
http://mingw.org
This is free software; see the product documentation or source code, for copying and redistribution conditions. There is NO WARRANTY; not even an implied WARRANTY OF MERCHANTABILITY, nor of FITNESS FOR ANY PARTICULAR PURPOSE.
This tool will guide you through the first time setup of the MinGW Installation Manager software (mingw-get) on your computer; additionally, it will offer you the opportunity to install some other common components of the MinGW software distribution.
After first time setup has been completed, you should invoke the MinGW Installation Manager directly, (either the CLI mingw-get.exe variant, or its GUI counterpart, according to your preference), when you wish to add or to remove components, or to upgrade your MinGW software installation.
View Licence
点这个
Install
Cancel
CSDN @DreamSuif

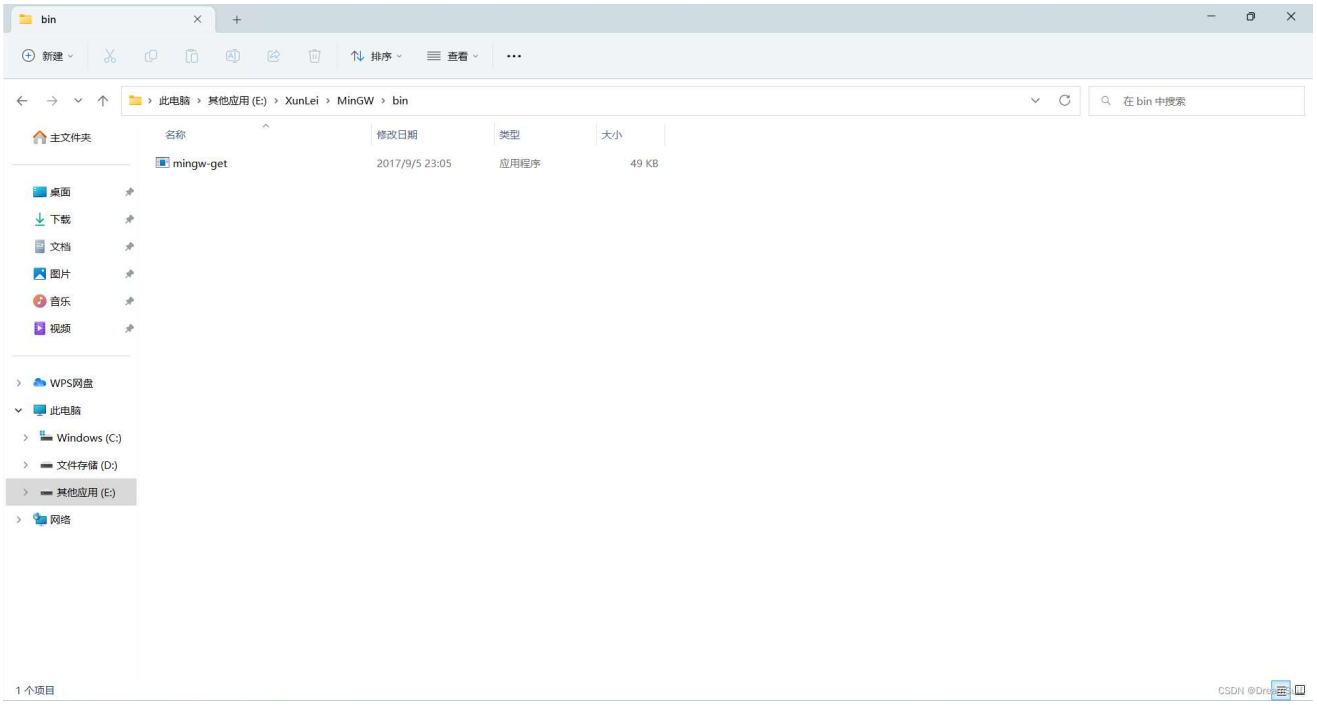

MinGW Installation Manager Setup Tool
mingw-get version 0.6.3-pre-20170905-1
1.点这里选择一个合适的安装位置，并记住这个位置，之后添加进系统环境的时候会用到
Step 1: Specify Installation Preferences
Installation Directory
C:\MinGW
Change
If you elect to change this, you are advised to avoid any choice of directory which includes white space within the absolute representation of its path name.
User Interface Options
Both command line and graphical options are available. The command line interface is always supported; the alternative only if you choose the following option to ...
... also install support for the graphical user interface.
Program shortcuts for launching the graphical user interface should be installed ...
... just for me (the current user), or ... ... for all users * ...
... in the start menu, and/or ... ... on the desktop.
* selection of this option requires administrative privilege.
2.然后点这里继续
View Licence
Continue
Cancel
CSDN @DreamSuif

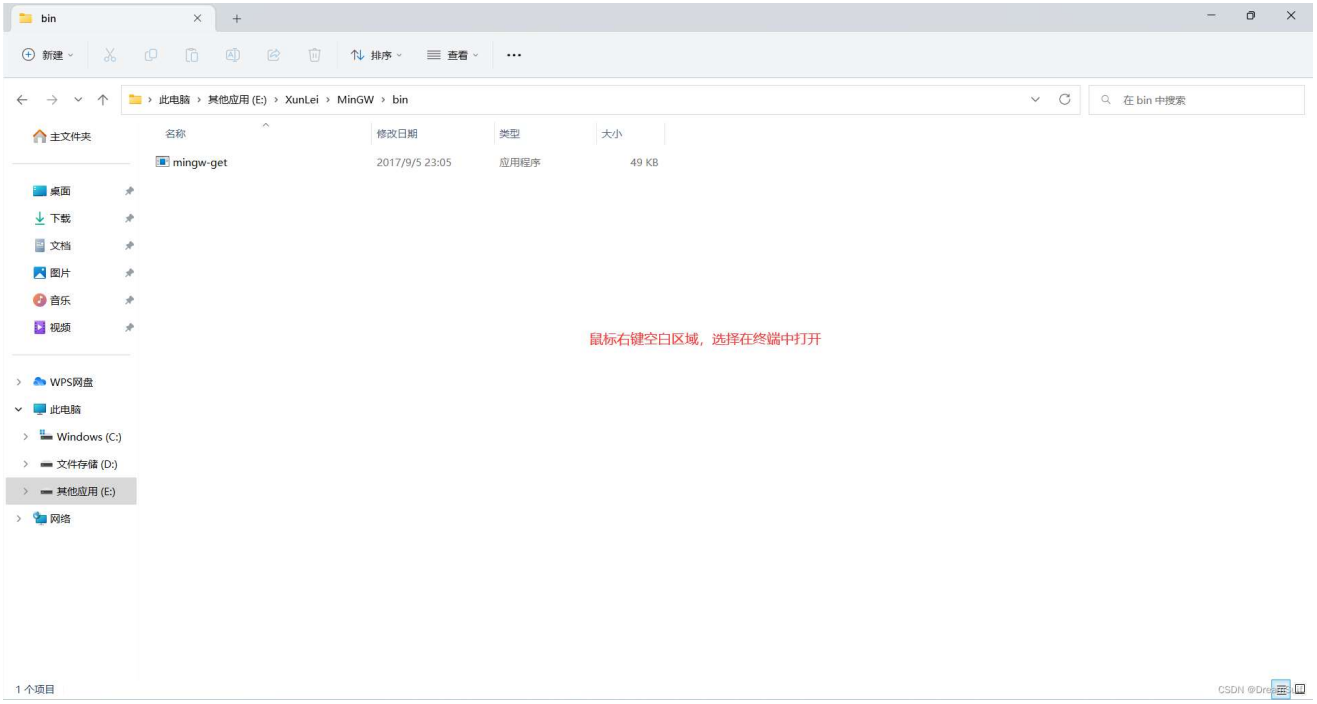

MinGW Installation Manager Setup Tool
mingw-get version 0.6.3-pre-20170905-1
Step 2: Download and Set Up MinGW Installation Manager
Download Progress
Connecting to osdn.net ... \
of
:
Details
静静等待安装完成
View Licence
Continue
Quit
CSDN @DreamSuif

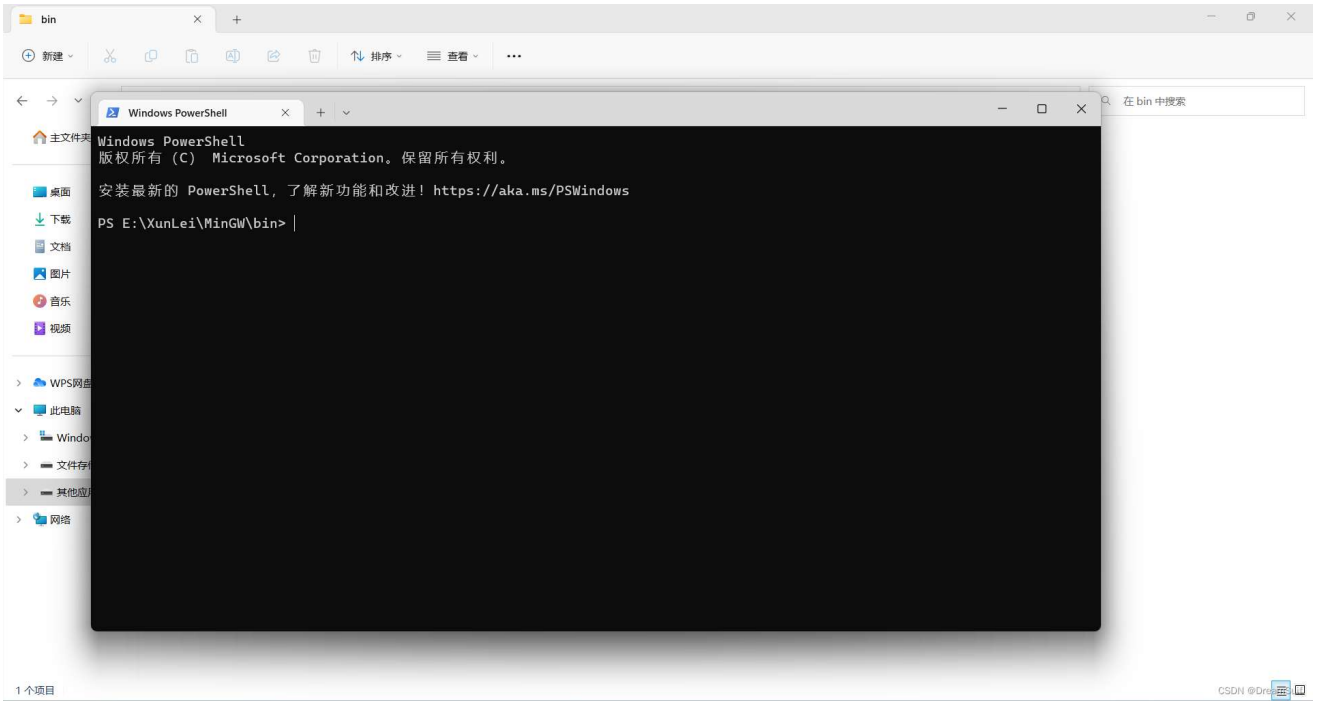


> 然后我们找到mingw的安装文件夹


MinGW
新建
排序
查看
此电脑 › 其他应用 (E:) › XunLei › MinGW
在 MinGW 中搜索
主文件夹
桌面
下载
文档
图片
音乐
视频
WPS网盘
此电脑
Windows (C:)
文件存储 (D:)
其他应用 (E:)
网络
名称
修改日期
类型
大小
bin 2022/11/12 9:48 文件夹
libexec 2022/11/12 9:48 文件夹
share 2022/11/12 9:48 文件夹
var 2022/11/12 9:48 文件夹
进入这个文件夹
4 个项目

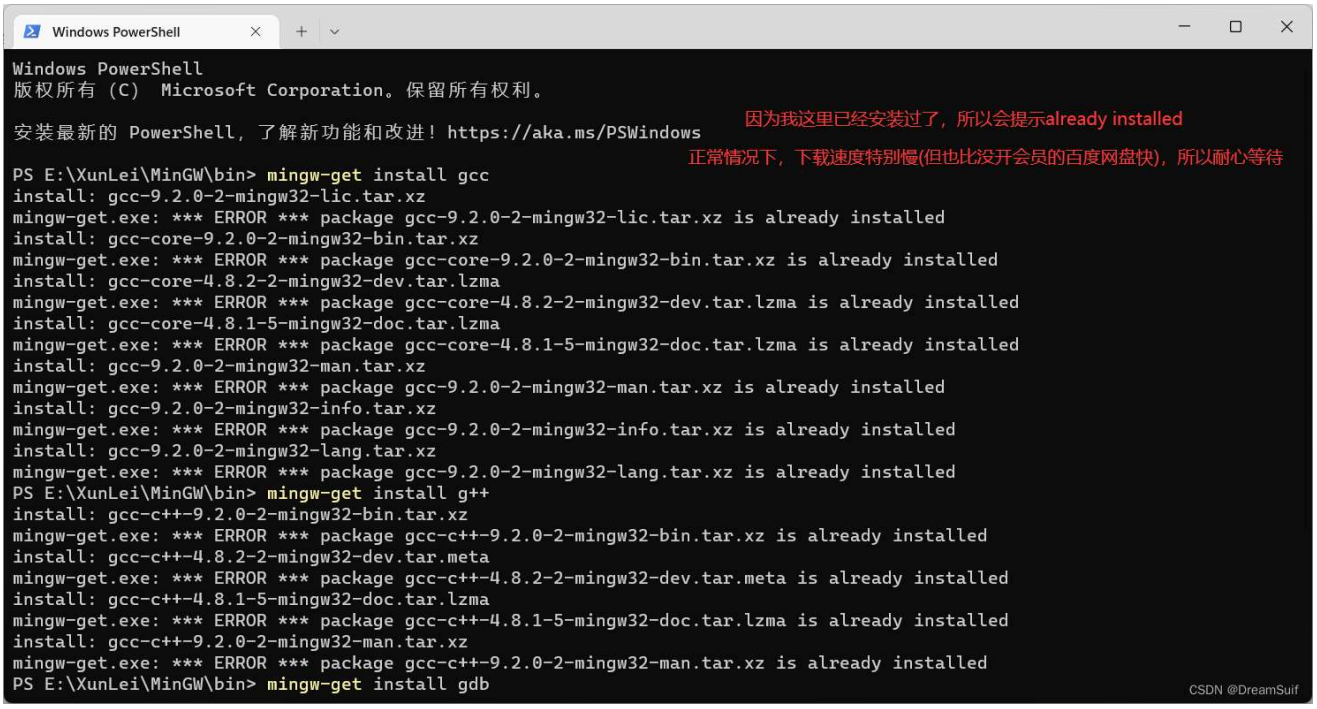

bin
新建
排序
查看
此电脑 › 其他应用 (E:) › XunLei › MinGW › bin
在 bin 中搜索
主文件夹
桌面
下载
文档
图片
音乐
视频
WPS网盘
此电脑
Windows (C:)
文件存储 (D:)
其他应用 (E:)
网络
名称
修改日期
类型
大小
mingw-get 2017/9/5 23:05 应用程序 49 KB
1 个项目

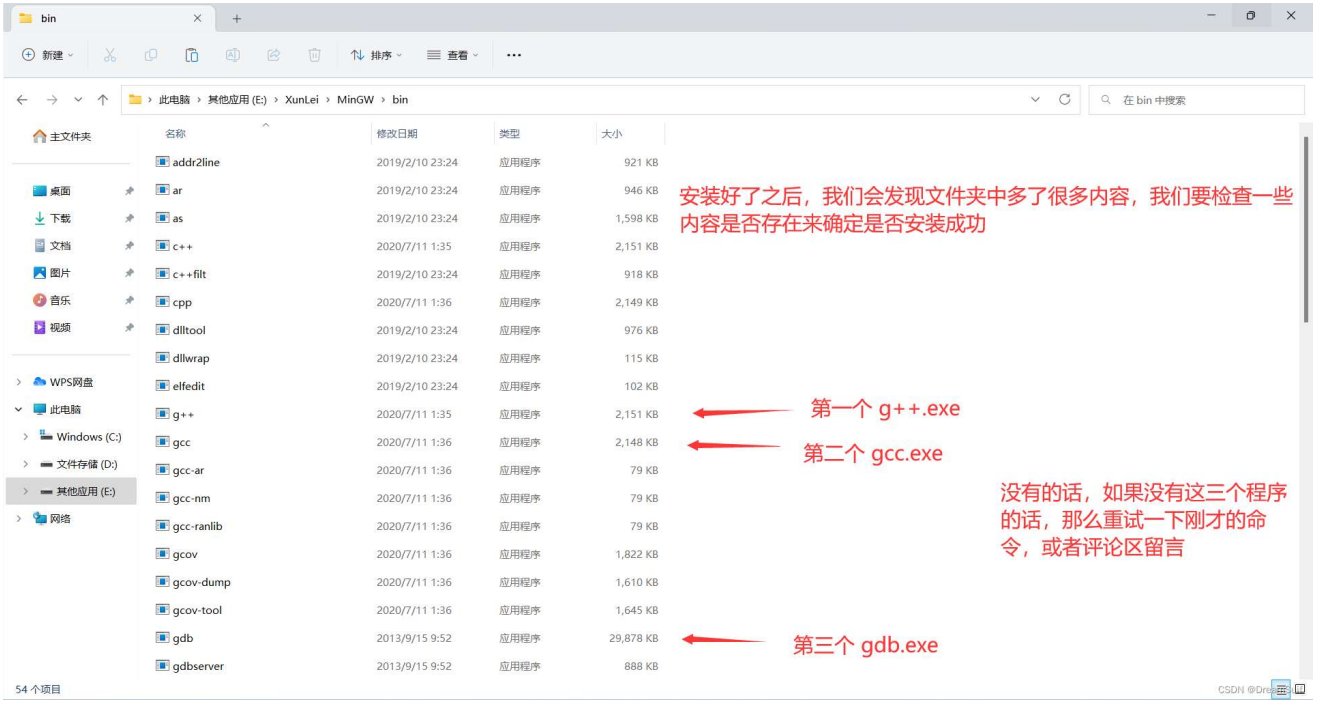




依次输入

```
mingw-get install gcc
mingw-get install g++
mingw-get install gdb
```

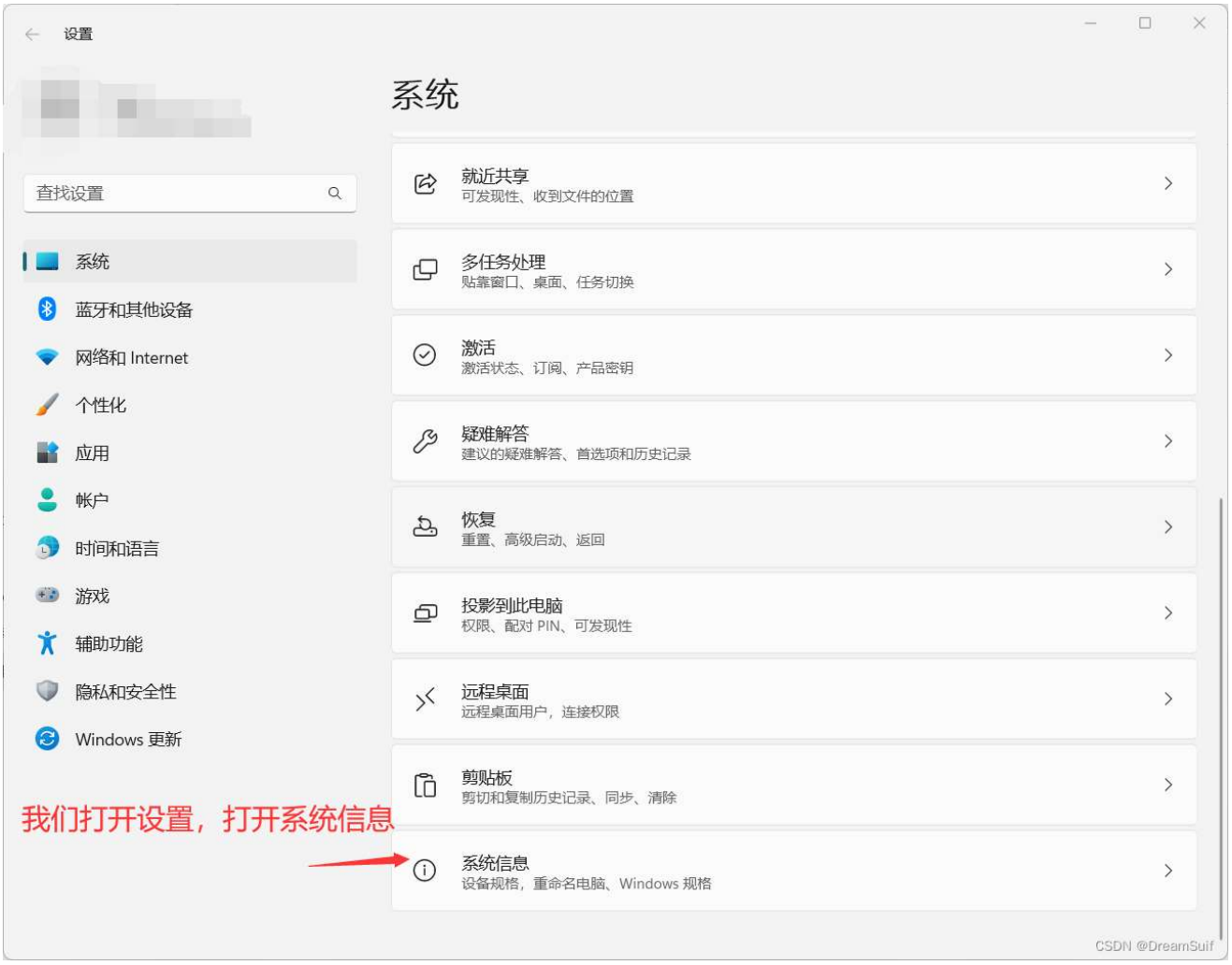




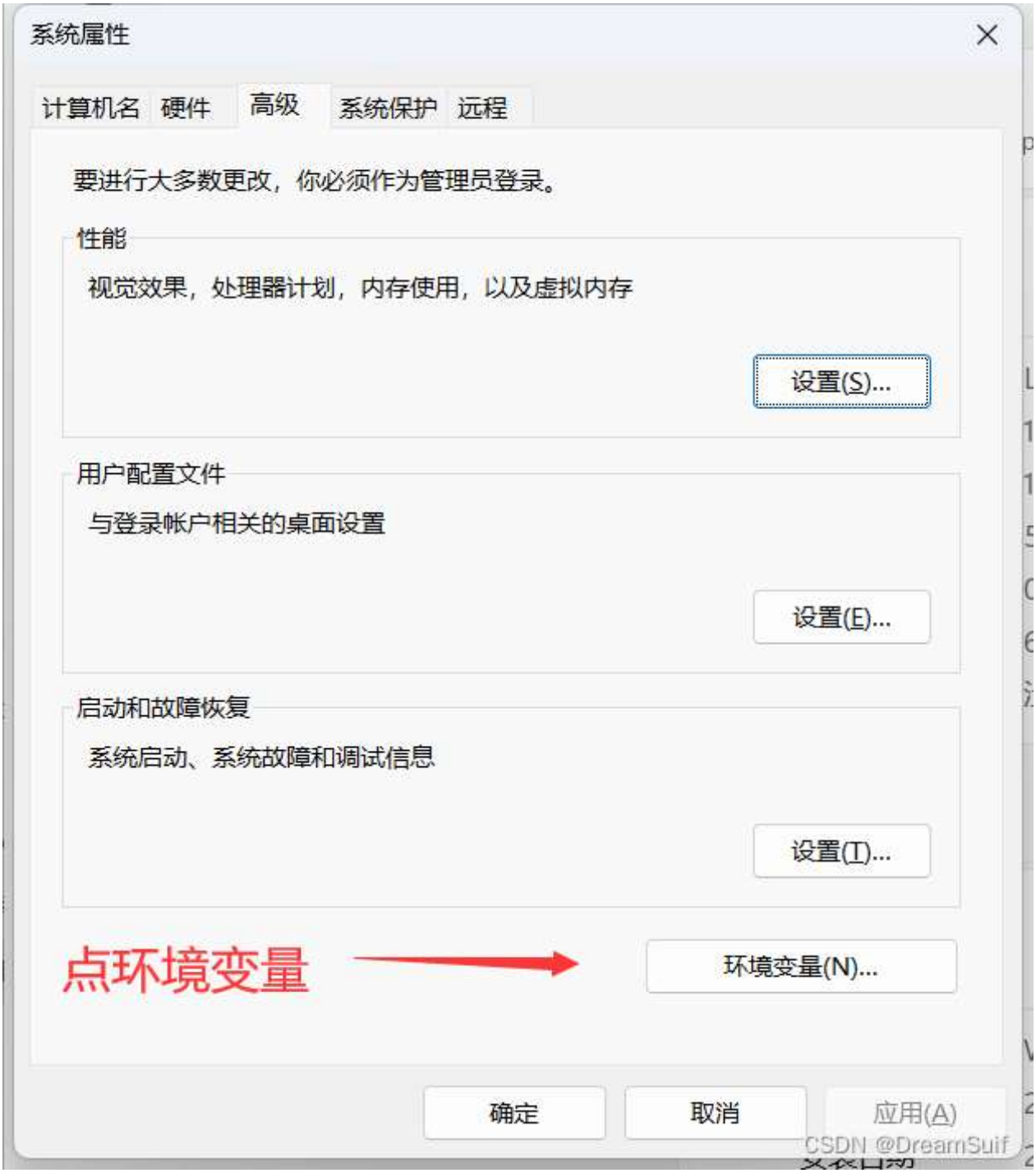
> 然后mingw就下载完成了

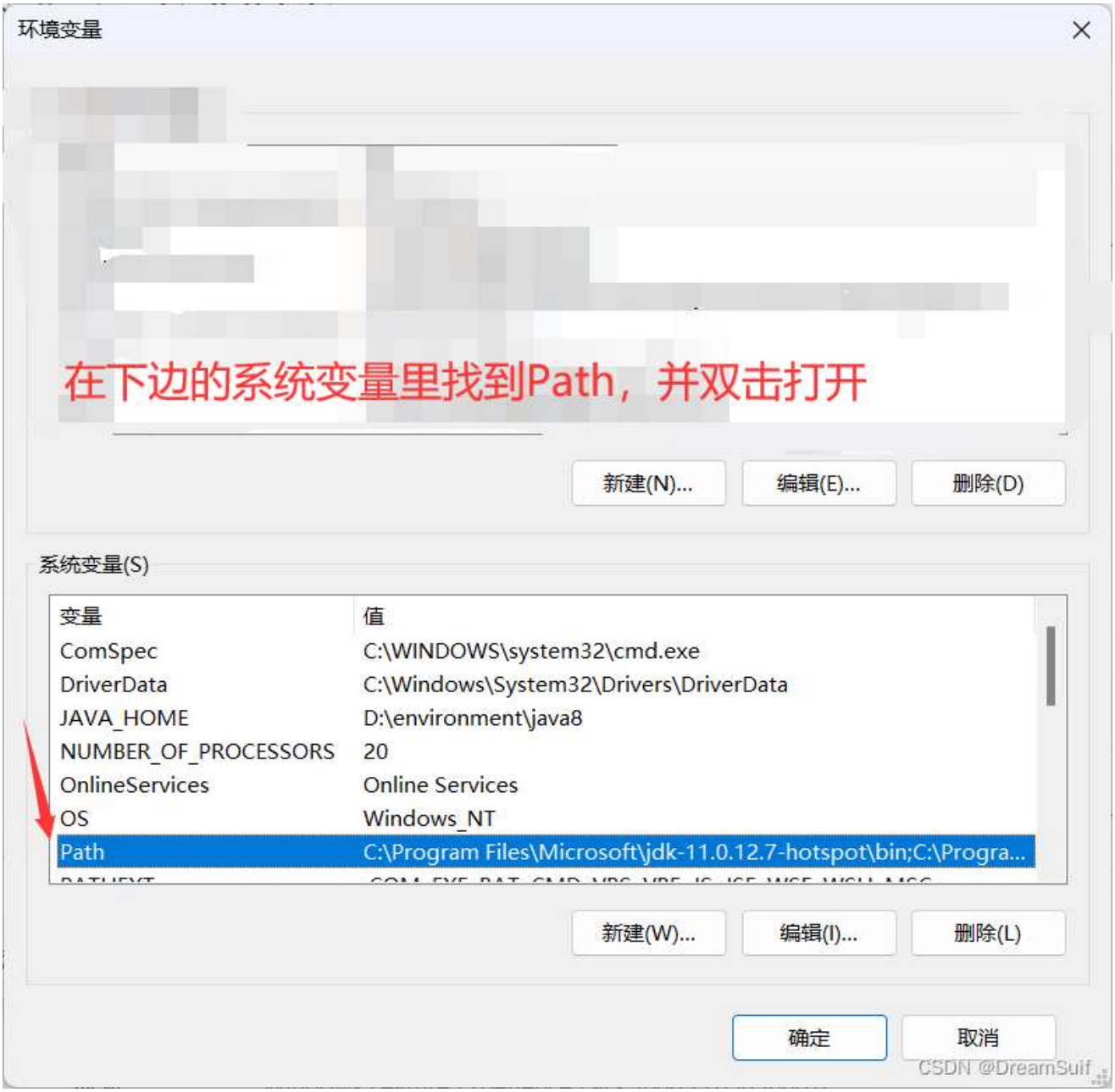
第三步 将mingw添加至系统变量中

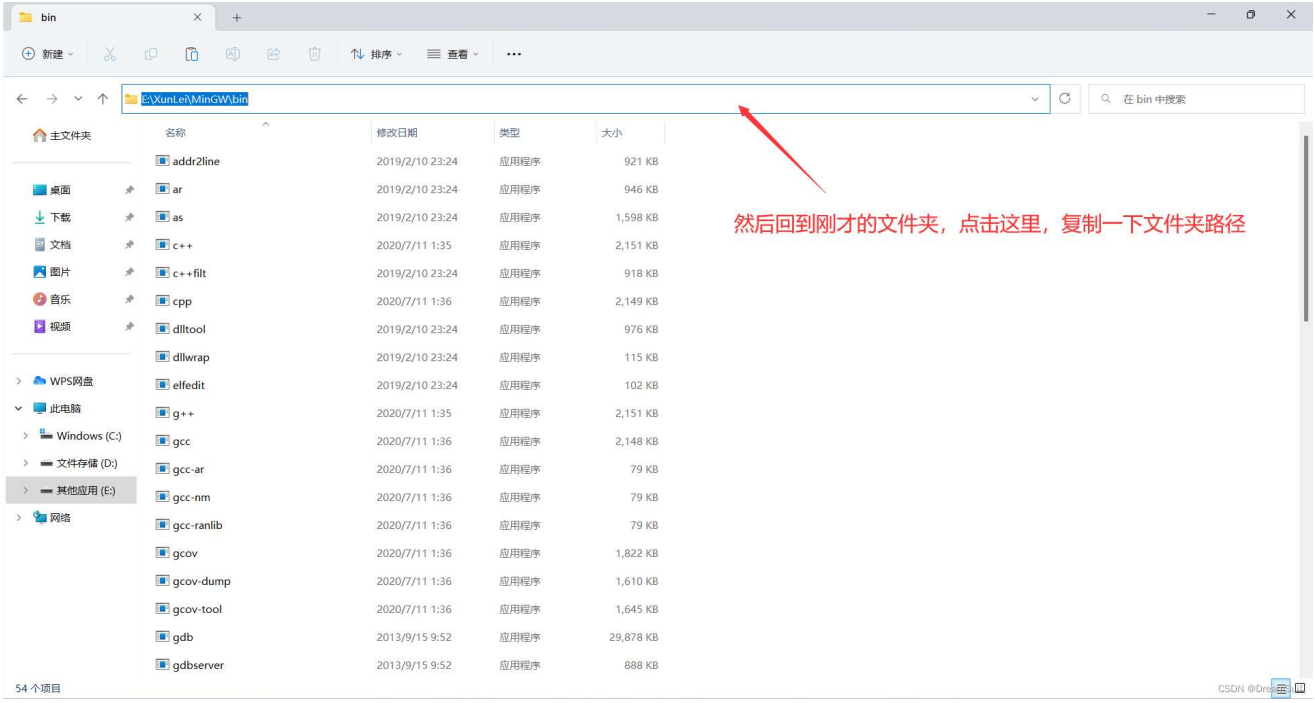
> 这一步就是告诉电脑，可以在哪个文件夹里找到 C/C++ 的运行环境

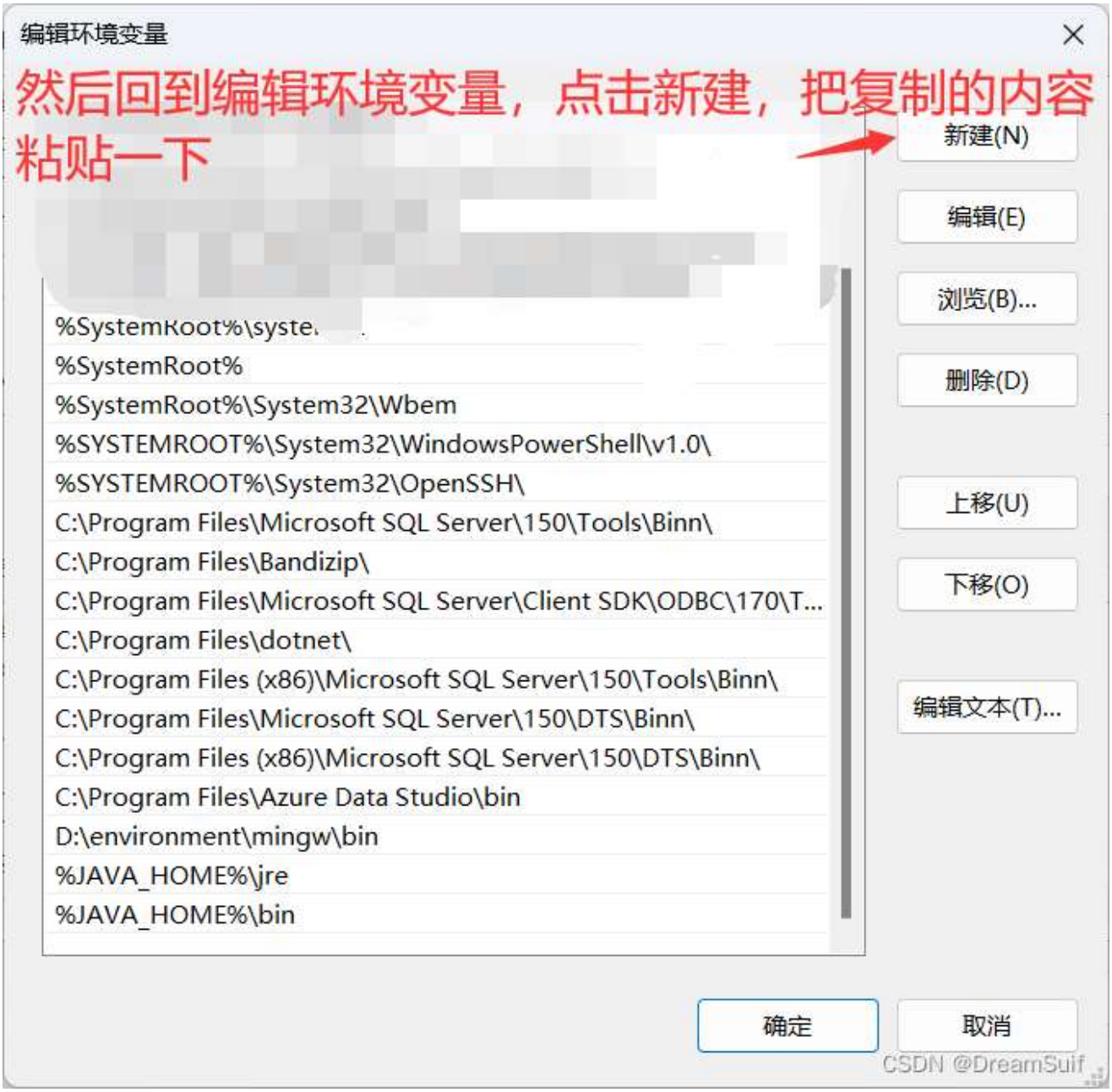

设置
系统
查找设置
系统
蓝牙和其他设备
网络和 Internet
个性化
应用
帐户
时间和语言
游戏
辅助功能
隐私和安全性
Windows 更新
就近共享
可发现性、收到文件的位置
多任务处理
贴靠窗口、桌面、任务切换
激活
激活状态、订阅、产品密钥
疑难解答
建议的疑难解答、首选项和历史记录
恢复
重置、高级启动、返回
投影到此电脑
权限、配对 PIN、可发现性
远程桌面
远程桌面用户，连接权限
剪贴板
剪切和复制历史记录、同步、清除
我们打开设置，打开系统信息
系统信息
设备规格，重命名电脑、Windows 规格
CSDN @DreamSuif

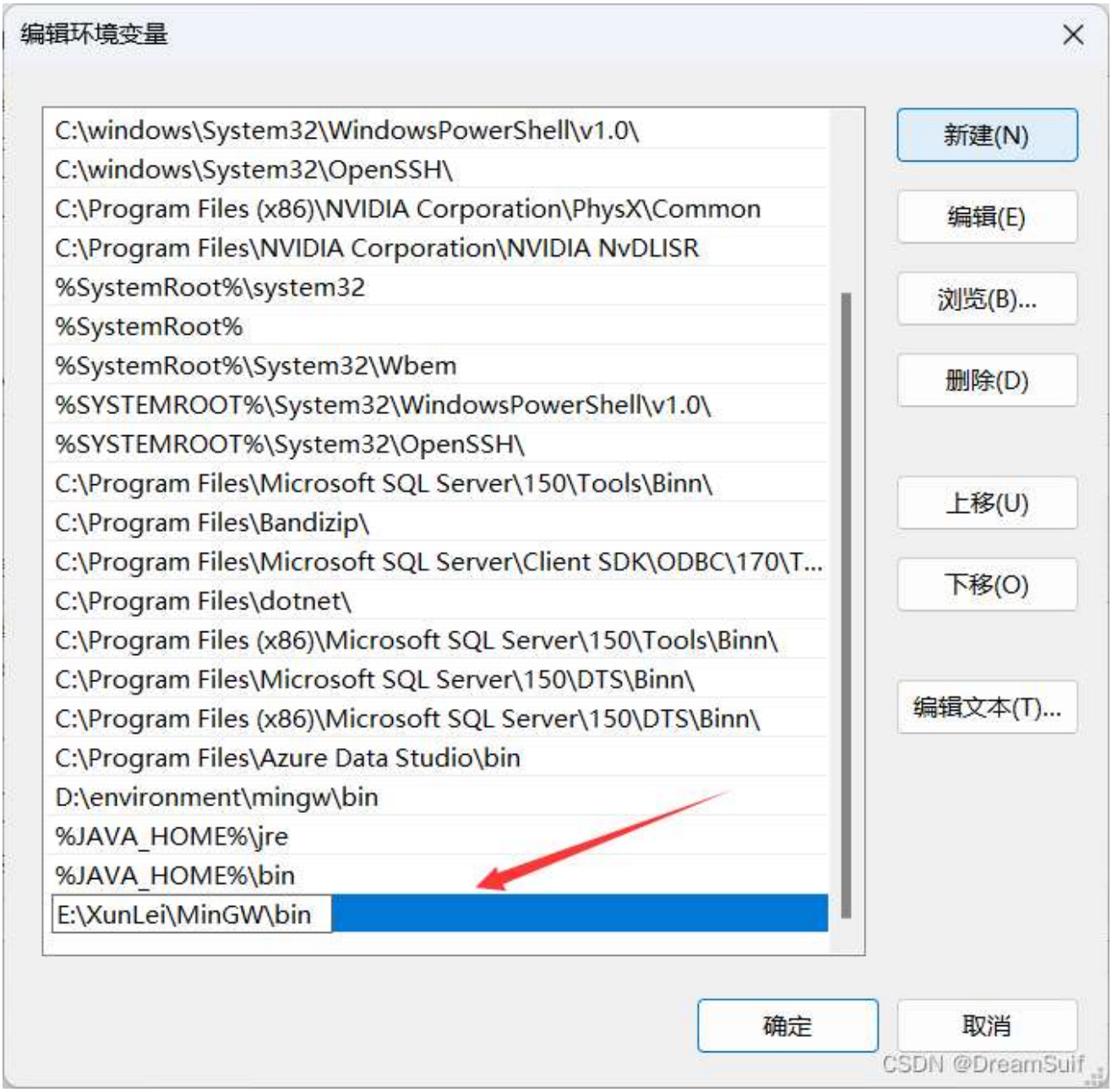

设置
系统 › 系统信息
查找设置
系统
蓝牙和其他设备
网络和 Internet
个性化
应用
帐户
时间和语言
游戏
辅助功能
隐私和安全性
Windows 更新
重命名这台电脑
设备规格
复制
处理器
12th Gen Intel(R) Core(TM) i7-12700H 2.30 GHz
机带 RAM
16.0 GB (15.7 GB 可用)
系统类型
64 位操作系统, 基于 x64 的处理器
笔和触控
没有可用于此显示器的笔或触控输入
点高级系统设置
相关链接
域或工作组
系统保护
高级系统设置
Windows 规格
复制
版本
Windows 11 家庭中文版
版本
22H2
安装日期
2022/11/11
操作系统版本
22621.819
Microsoft 服务协议
Microsoft 软件许可条款
CSDN @DreamSuif

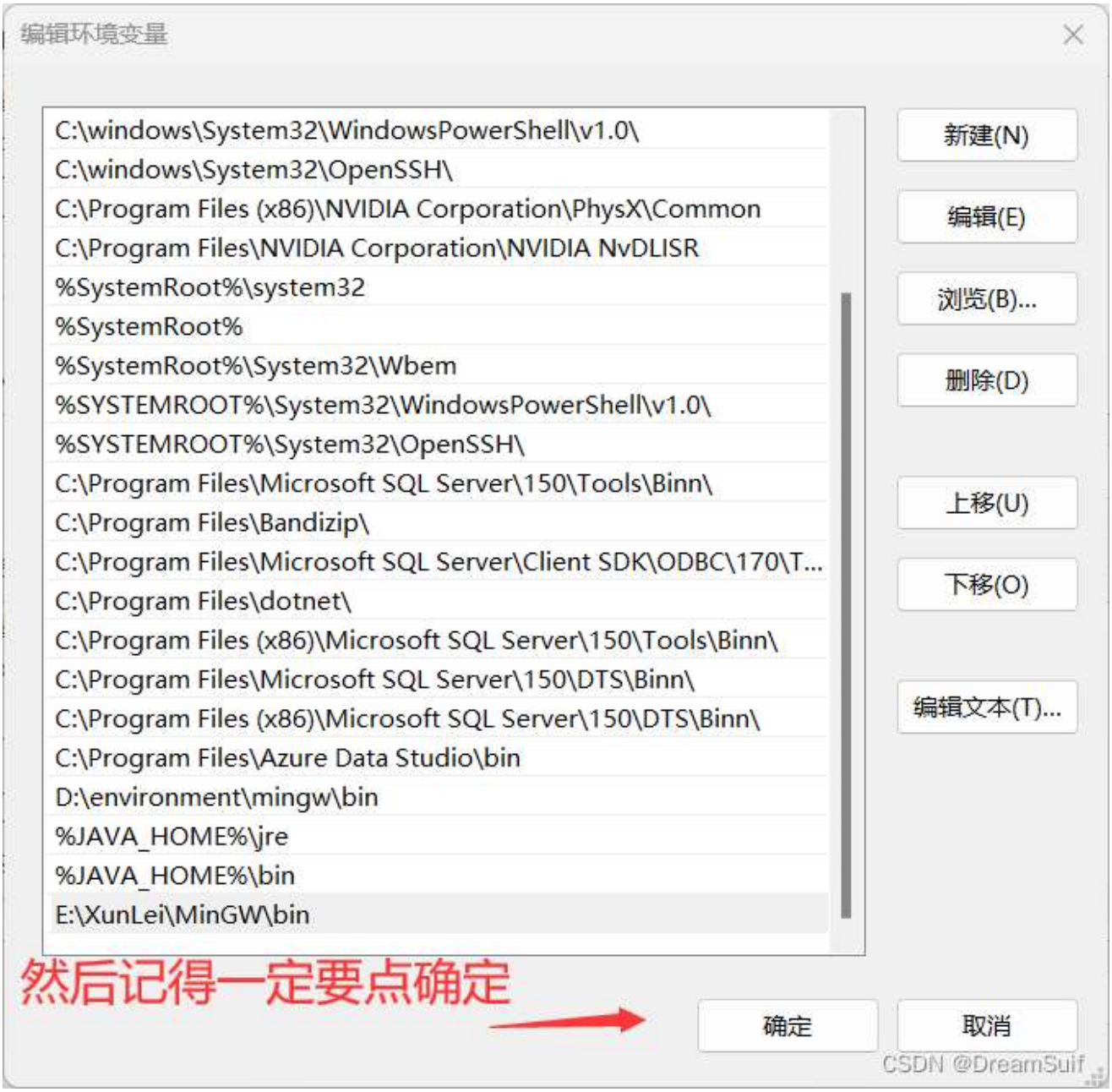

系统属性
计算机名
硬件
高级
系统保护
远程
要进行大多数更改，你必须作为管理员登录。
性能
视觉效果，处理器计划，内存使用，以及虚拟内存
设置(S)...
用户配置文件
与登录帐户相关的桌面设置
设置(E)...
启动和故障恢复
系统启动、系统故障和调试信息
设置(T)...
点环境变量
环境变量(N)...
确定
取消
应用(A)
CSDN @DreamSuif

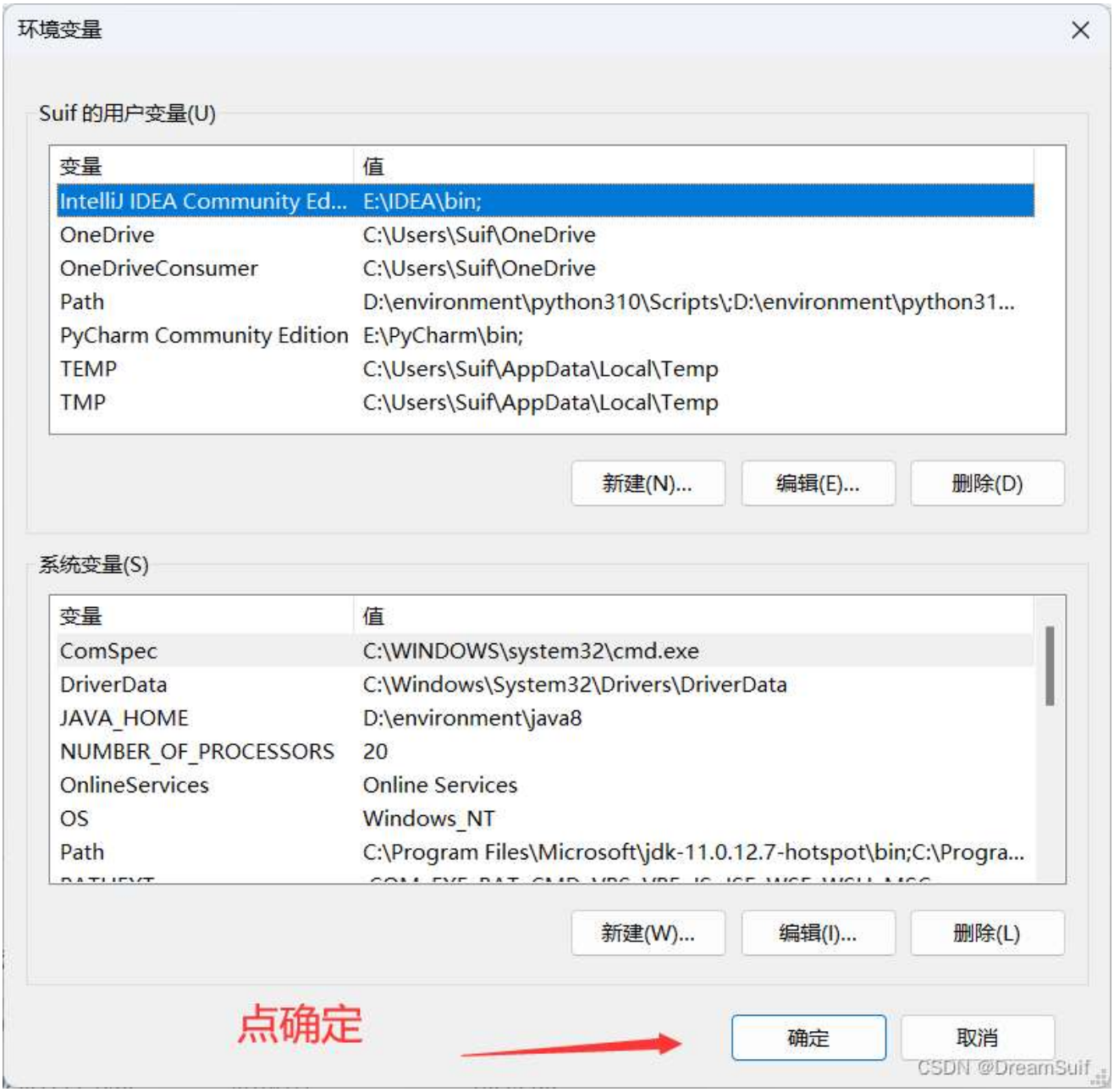

环境变量
在下边的系统变量里找到Path，并双击打开
新建(N)...
编辑(E)...
删除(D)
系统变量(S)
变量
值
ComSpec
C:\WINDOWS\system32\cmd.exe
DriverData
C:\Windows\System32\Drivers\DriverData
JAVA_HOME
D:\environment\java8
NUMBER_OF_PROCESSORS
20
OnlineServices
Online Services
OS
Windows_NT
Path
C:\Program Files\Microsoft\jdk-11.0.12.7-hotspot\bin;C:\Progra...
新建(W)...
编辑(I)...
删除(L)
确定
取消
CSDN @DreamSuif

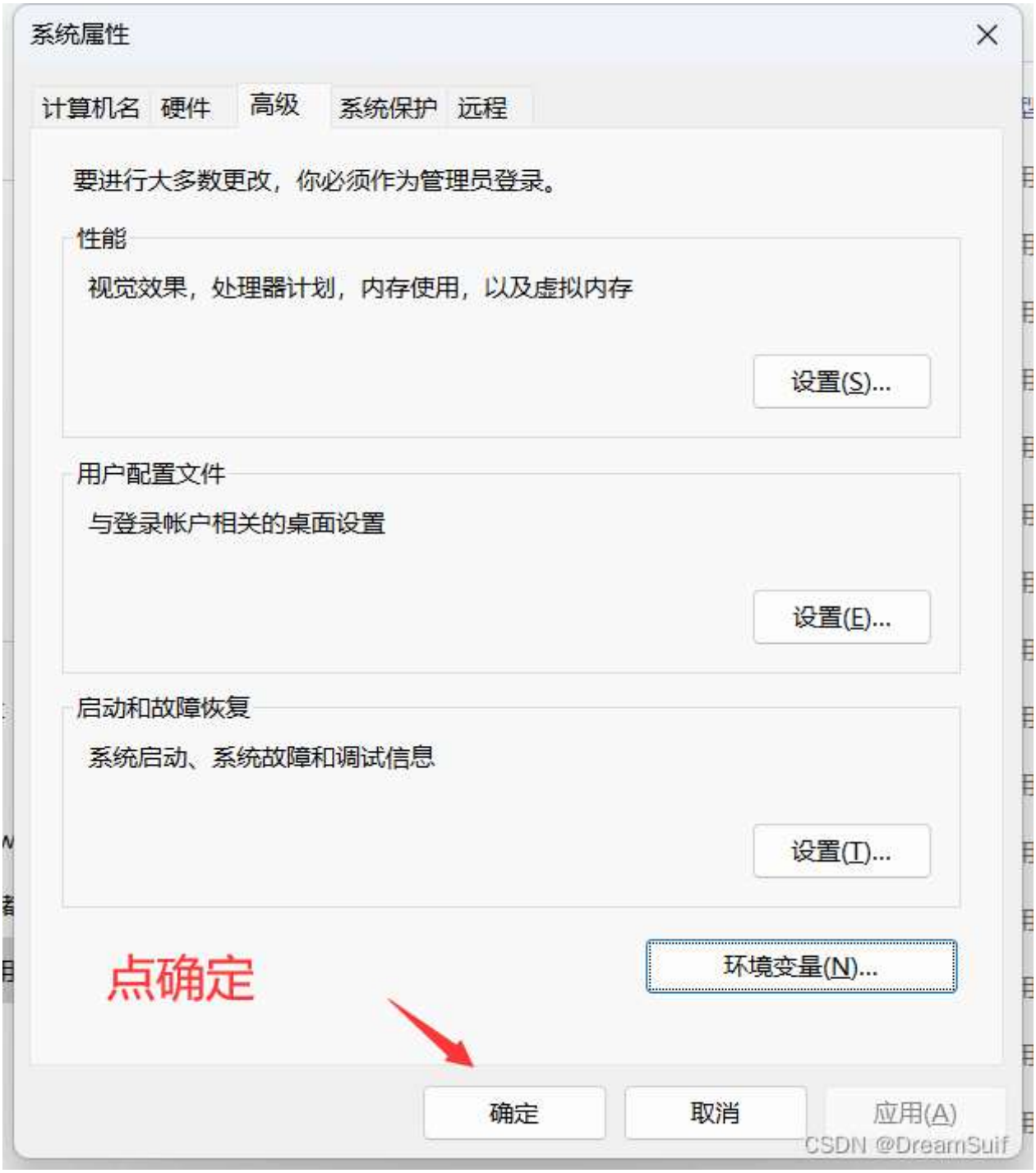

bin
新建
排序
查看
E:\XunLei\MinGW\bin
在 bin 中搜索
主文件夹
桌面
下载
文档
图片
音乐
视频
WPS网盘
此电脑
Windows (C:)
文件存储 (D:)
其他应用 (E:)
网络
名称
修改日期
类型
大小
addr2line 2019/2/10 23:24 应用程序 921 KB
ar 2019/2/10 23:24 应用程序 946 KB
as 2019/2/10 23:24 应用程序 1,598 KB
c++ 2020/7/11 1:35 应用程序 2,151 KB
c++filt 2019/2/10 23:24 应用程序 918 KB
cpp 2020/7/11 1:36 应用程序 2,149 KB
dlltool 2019/2/10 23:24 应用程序 976 KB
dllwrap 2019/2/10 23:24 应用程序 115 KB
elfedit 2019/2/10 23:24 应用程序 102 KB
g++ 2020/7/11 1:35 应用程序 2,151 KB
gcc 2020/7/11 1:36 应用程序 2,148 KB
gcc-ar 2020/7/11 1:36 应用程序 79 KB
gcc-nm 2020/7/11 1:36 应用程序 79 KB
gcc-ranlib 2020/7/11 1:36 应用程序 79 KB
gcov 2020/7/11 1:36 应用程序 1,822 KB
gcov-dump 2020/7/11 1:36 应用程序 1,610 KB
gcov-tool 2020/7/11 1:36 应用程序 1,645 KB
gdb 2013/9/15 9:52 应用程序 29,878 KB
gdbserver 2013/9/15 9:52 应用程序 888 KB
54 个项目
然后回到刚才的文件夹，点击这里，复制一下文件夹路径


编辑环境变量
然后回到编辑环境变量，点击新建，把复制的内容粘贴一下
新建(N)
编辑(E)
浏览(B)...
删除(D)
上移(U)
下移(O)
编辑文本(T)...
%SystemRoot%\syste
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files\Bandizip\
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\T...
C:\Program Files\dotnet\
C:\Program Files (x86)\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files\Microsoft SQL Server\150\DTS\Binn\
C:\Program Files (x86)\Microsoft SQL Server\150\DTS\Binn\
C:\Program Files\Azure Data Studio\bin
D:\environment\mingw\bin
%JAVA_HOME%\jre
%JAVA_HOME%\bin
确定
取消
CSDN @DreamSuif


编辑环境变量
C:\windows\System32\WindowsPowerShell\v1.0\
C:\windows\System32\OpenSSH\
C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common
C:\Program Files\NVIDIA Corporation\NVIDIA NvDLISR
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files\Bandizip\
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\T...
C:\Program Files\dotnet\
C:\Program Files (x86)\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files\Microsoft SQL Server\150\DTS\Binn\
C:\Program Files (x86)\Microsoft SQL Server\150\DTS\Binn\
C:\Program Files\Azure Data Studio\bin
D:\environment\mingw\bin
%JAVA_HOME%\jre
%JAVA_HOME%\bin
E:\XunLei\MinGW\bin
新建(N)
编辑(E)
浏览(B)...
删除(D)
上移(U)
下移(O)
编辑文本(T)...
确定
取消
CSDN @DreamSuif

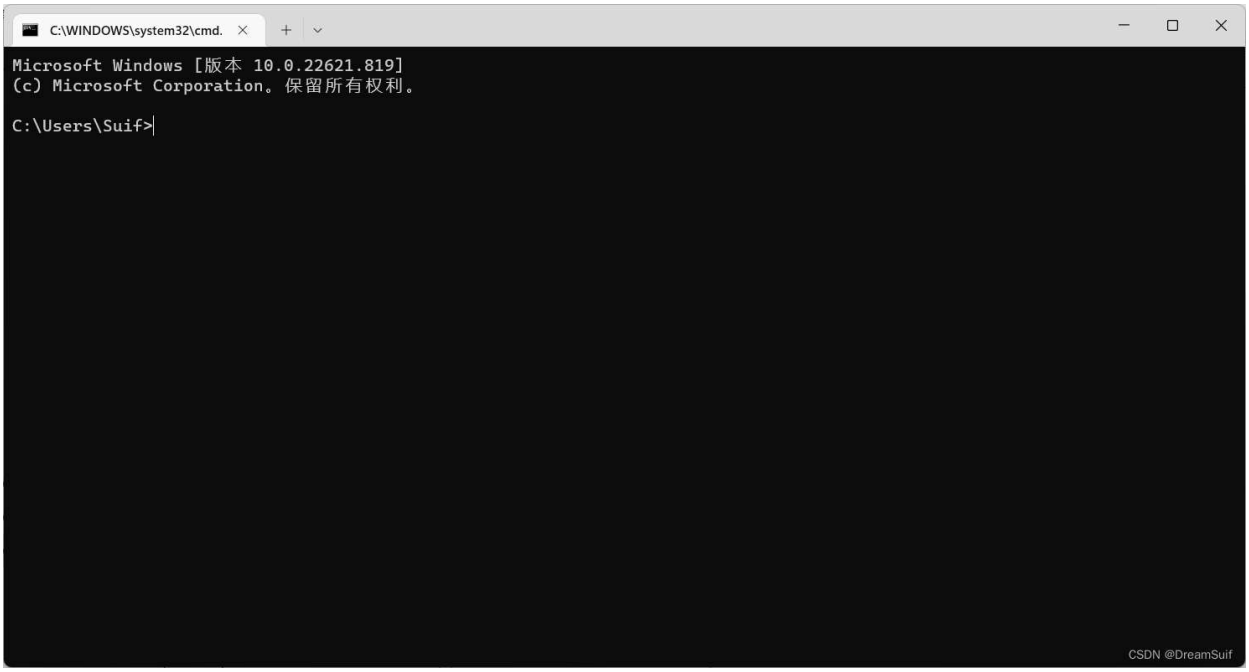

编辑环境变量
C:\windows\System32\WindowsPowerShell\v1.0\
C:\windows\System32\OpenSSH\
C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common
C:\Program Files\NVIDIA Corporation\NVIDIA NvDLISR
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files\Bandizip\
C:\Program Files\Microsoft SQL Server\Client SDK\ODBC\170\T...
C:\Program Files\dotnet\
C:\Program Files (x86)\Microsoft SQL Server\150\Tools\Binn\
C:\Program Files\Microsoft SQL Server\150\DTS\Binn\
C:\Program Files (x86)\Microsoft SQL Server\150\DTS\Binn\
C:\Program Files\Azure Data Studio\bin
D:\environment\mingw\bin
%JAVA_HOME%\jre
%JAVA_HOME%\bin
E:\XunLei\MinGW\bin
新建(N)
编辑(E)
浏览(B)...
删除(D)
上移(U)
下移(O)
编辑文本(T)...
然后记得一定要点确定
确定
取消
CSDN @DreamSuif


环境变量
Suif 的用户变量(U)
变量
值
IntelliJ IDEA Community Ed...
E:\IDEA\bin;
OneDrive
C:\Users\Suif\OneDrive
OneDriveConsumer
C:\Users\Suif\OneDrive
Path
D:\environment\python310\Scripts\;D:\environment\python31...
PyCharm Community Edition
E:\PyCharm\bin;
TEMP
C:\Users\Suif\AppData\Local\Temp
TMP
C:\Users\Suif\AppData\Local\Temp
新建(N)...
编辑(E)...
删除(D)
系统变量(S)
变量
值
ComSpec
C:\WINDOWS\system32\cmd.exe
DriverData
C:\Windows\System32\Drivers\DriverData
JAVA_HOME
D:\environment\java8
NUMBER_OF_PROCESSORS
20
OnlineServices
Online Services
OS
Windows_NT
Path
C:\Program Files\Microsoft\jdk-11.0.12.7-hotspot\bin;C:\Progra...
新建(W)...
编辑(I)...
删除(L)
点确定
确定
取消
CSDN @DreamSuif



这样我们的系统环境就配置完成了
我们可以来检验一下是否配置正确
按住 win + r 打开以下界面

输入 cmd， 然后点确定



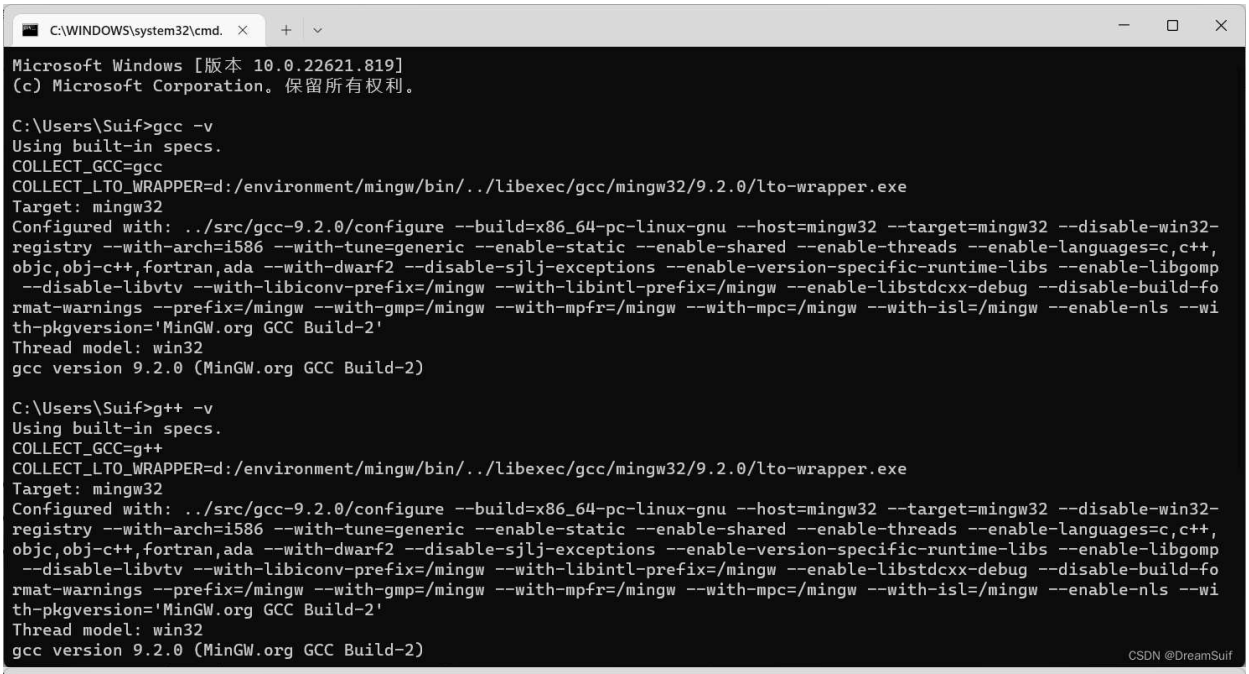
依次输入

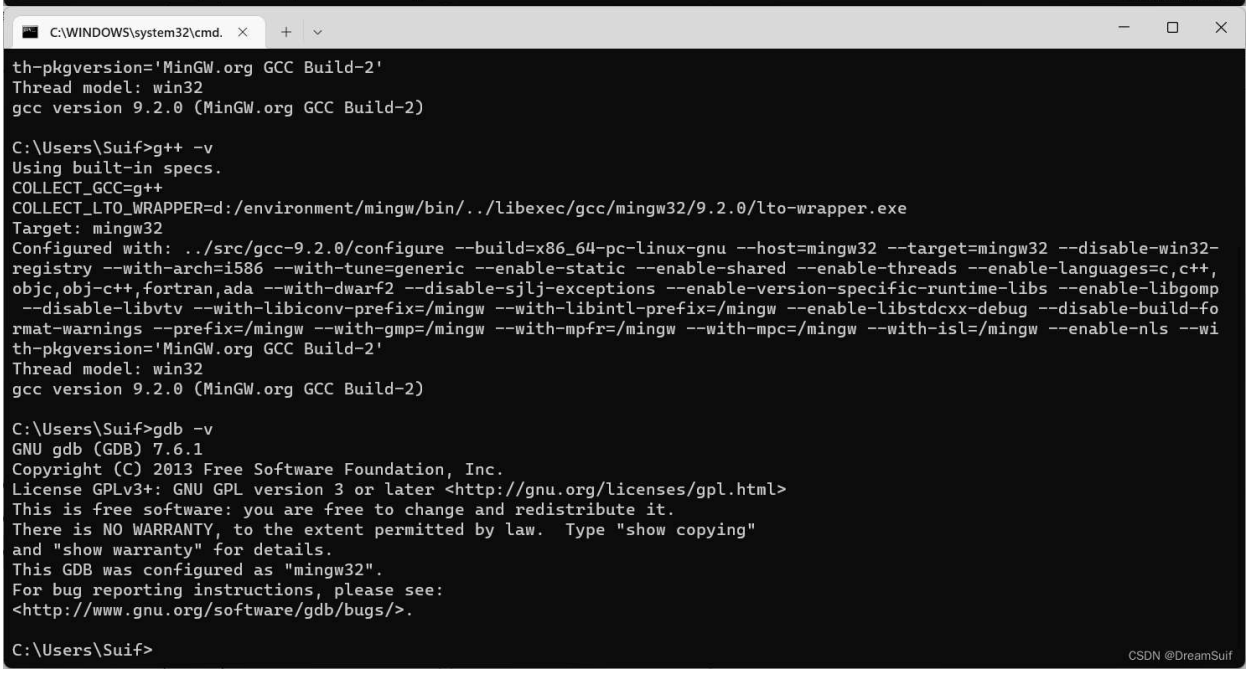
gcc -v

g++ -v

gdb -v

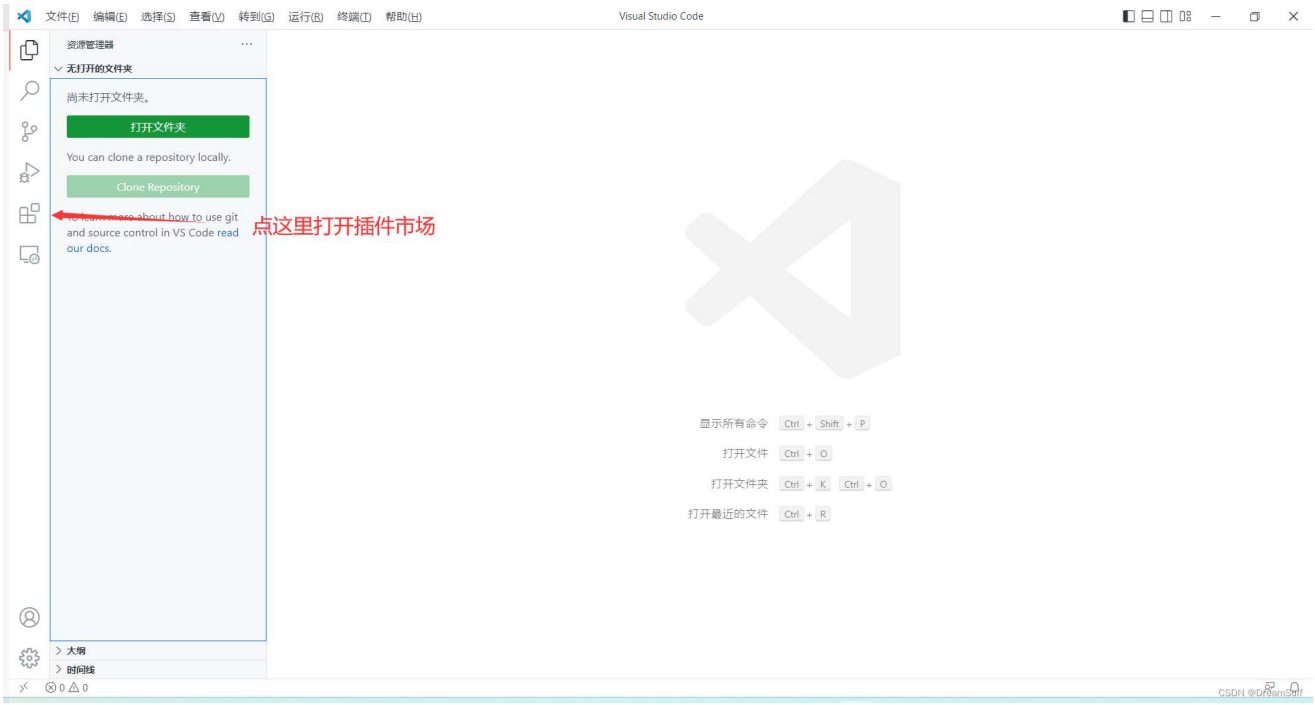
看看是否显示正常结果





结果正常显示就表示系统环境配置正确，这时候建议重启一下电脑(当然也可以不重启)

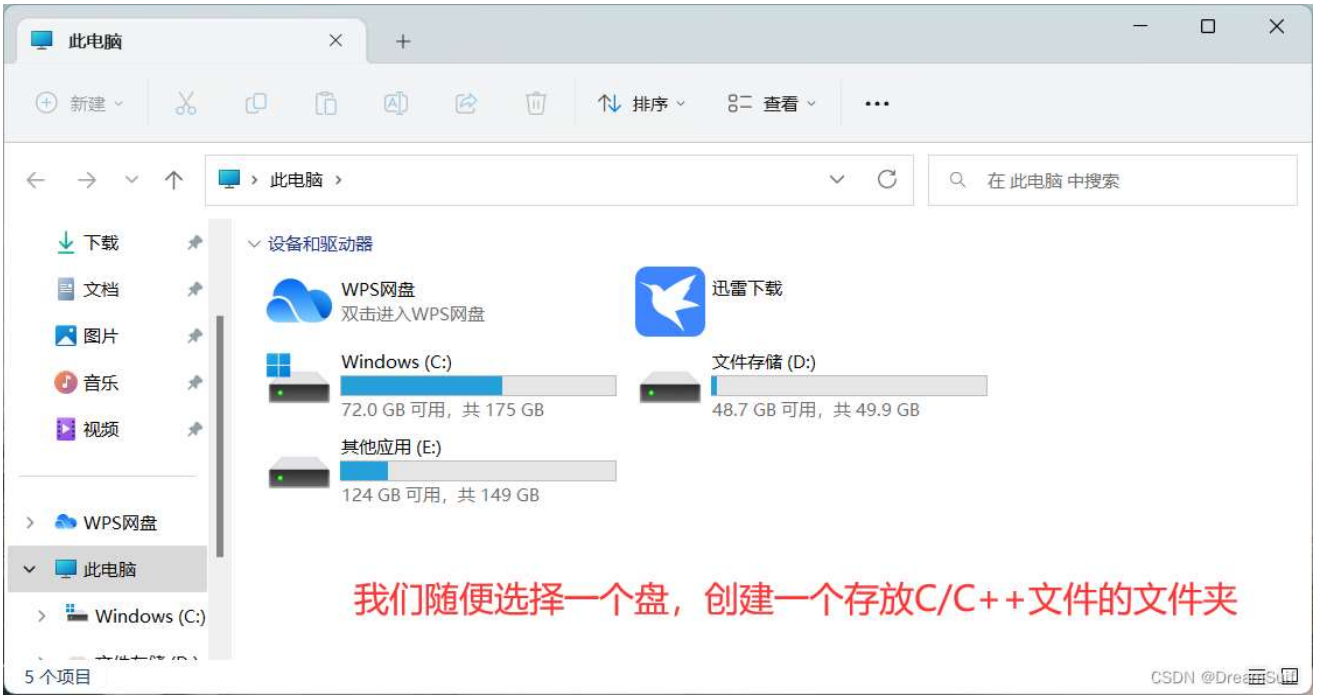
第四步 打开VsCode安装一下必要的插件

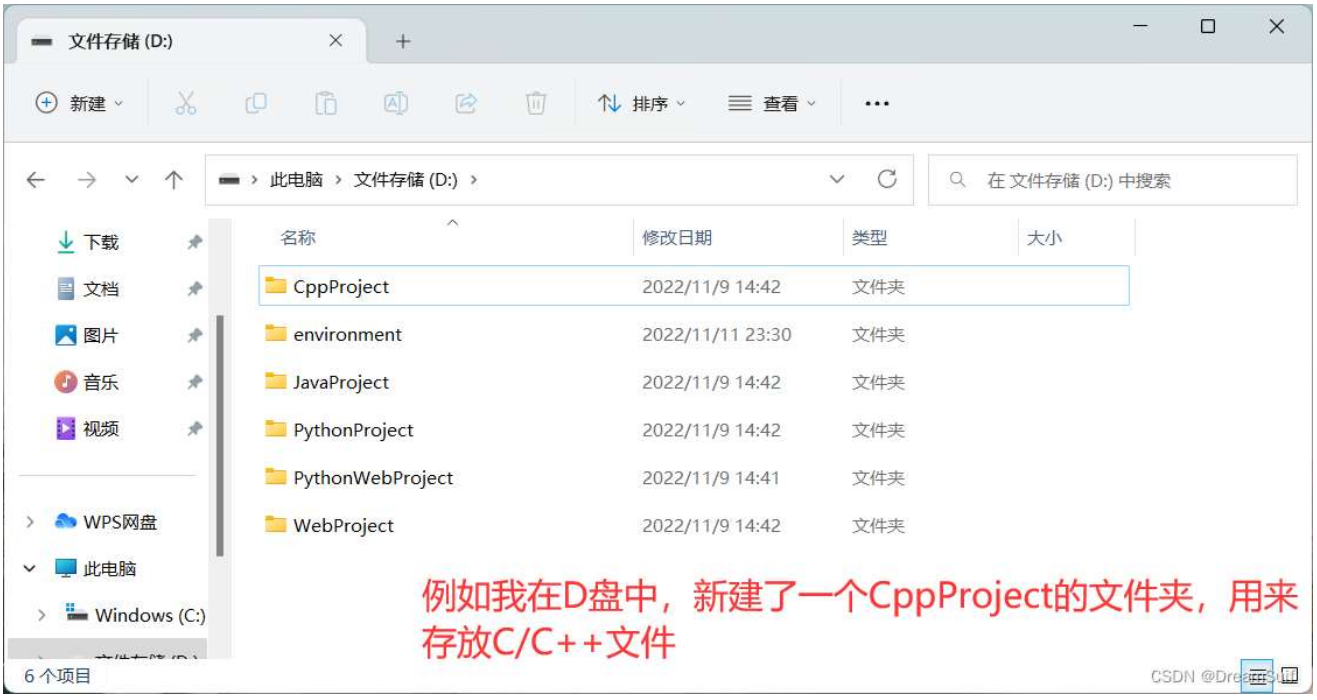


> 我们的vscode配置 C/C++ 的准备工作就完成了

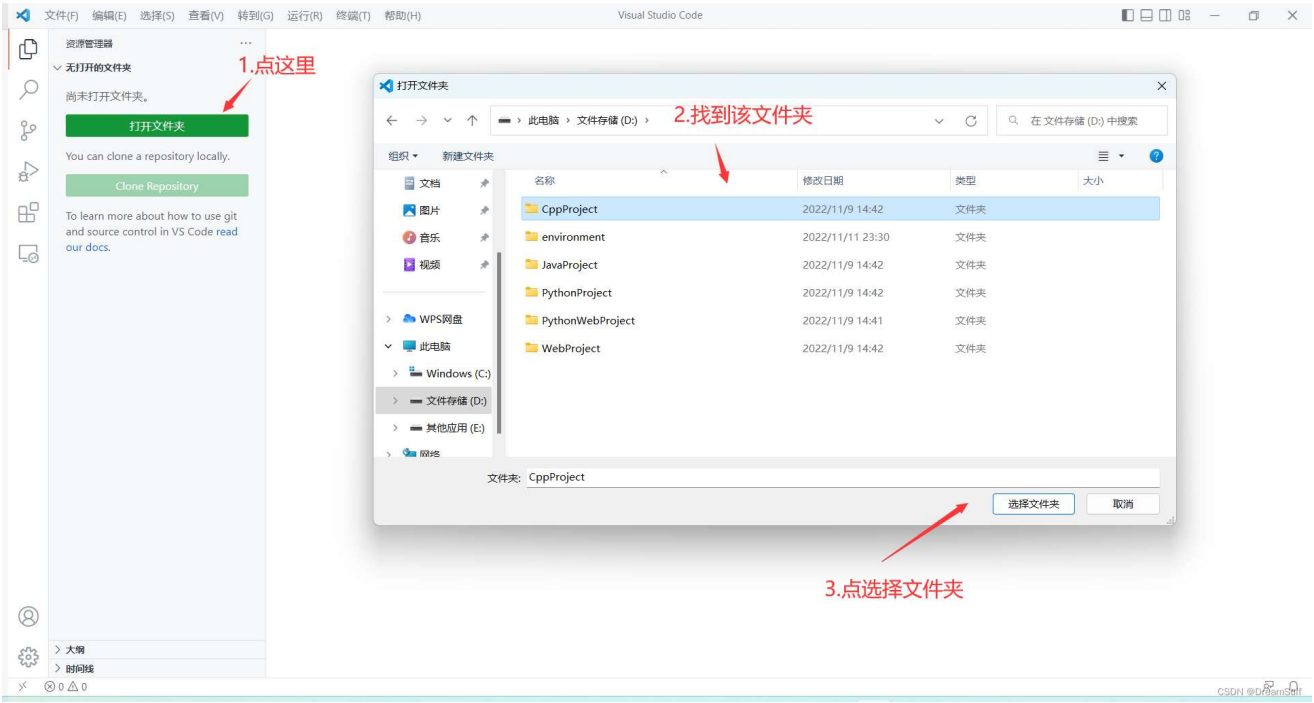
正式开始配置

第一步 新建个存放C/C++文件的文件夹，并新建个cpp文件

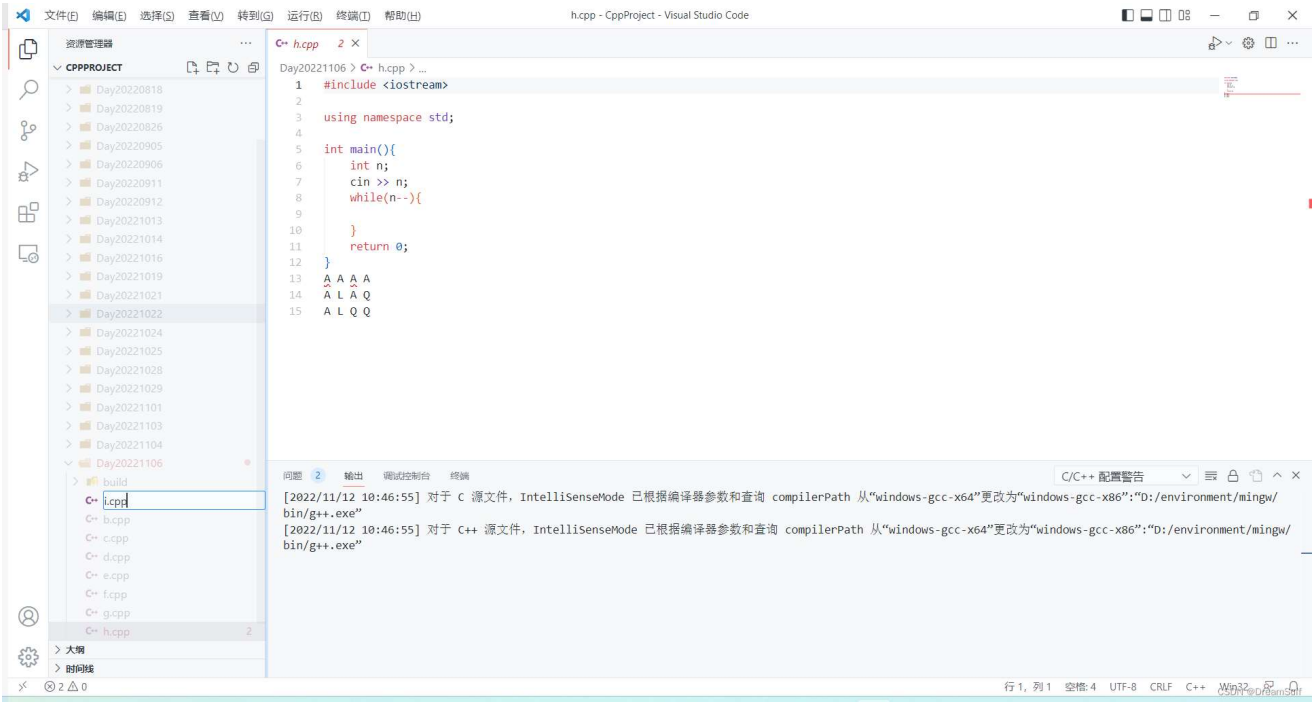


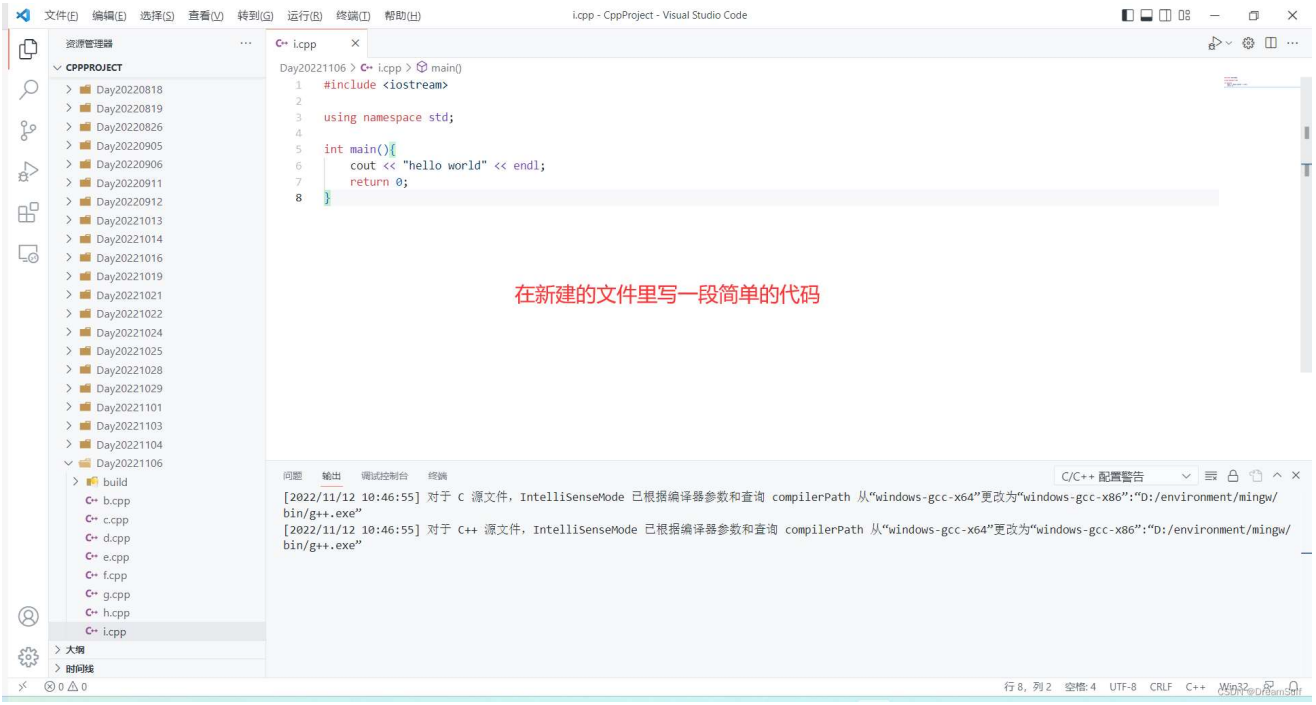


> 我们在vscode中打开那个文件夹


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H)
Visual Studio Code
资源管理器
无打开的文件夹
1.点这里
尚未打开文件夹。
打开文件夹
You can clone a repository locally.
Clone Repository
To learn more about how to use git and source control in VS Code read our docs.
大纲
时间线
打开文件夹
此电脑 › 文件存储 (D:) ›
2.找到该文件夹
在 文件存储 (D:) 中搜索
组织
新建文件夹
文档
图片
音乐
视频
WPS网盘
此电脑
Windows (C:)
文件存储 (D:)
其他应用 (E:)
网络
名称
修改日期
类型
大小
CppProject 2022/11/9 14:42 文件夹
environment 2022/11/11 23:30 文件夹
JavaProject 2022/11/9 14:42 文件夹
PythonProject 2022/11/9 14:42 文件夹
PythonWebProject 2022/11/9 14:41 文件夹
WebProject 2022/11/9 14:42 文件夹
文件夹: CppProject
选择文件夹
取消
3.点选择文件夹
CSDN @DreamSoft


文件(F) 编辑(E) 选择(S) 查看(V) 转到(G) 运行(R) 终端(T) 帮助(H)
CppProject - Visual Studio Code
资源管理器
点这里，新建一个.cpp文件
CPPPROJECT
点这个可以在文件夹里新建一个文件夹
通讯录管理系统
Day20220818
Day20220819
Day20220826
Day20220905
Day20220906
Day20220911
Day20220912
Day20221013
Day20221014
Day20221016
Day20221019
Day20221021
Day20221022
Day20221024
Day20221025
Day20221028
Day20221029
Day20221101
Day20221103
Day20221104
Day20221106
build
b.cpp
c.cpp
d.cpp
e.cpp
f.cpp
g.cpp
h.cpp
我这里为了记录每天写的代码，所以我建了很多文件夹，存放每天写的代码
显示所有命令 Ctrl + Shift + P
转到文件 Ctrl + P
在文件中查找 Ctrl + Shift + F
开始调试 F5
切换终端 Ctrl + `
问题 输出 调试控制台 终端
C/C++ 配置警告
[2022/11/12 10:46:55] 对于 C 源文件，IntelliSenseMode 已根据编译器参数和查询 compilerPath 从“windows-gcc-x64”更改为“windows-gcc-x86”:“D:/environment/mingw/bin/g++.exe”
[2022/11/12 10:46:55] 对于 C++ 源文件，IntelliSenseMode 已根据编译器参数和查询 compilerPath 从“windows-gcc-x64”更改为“windows-gcc-x86”:“D:/environment/mingw/bin/g++.exe”
大纲
时间线
CSDN @DreamSoft

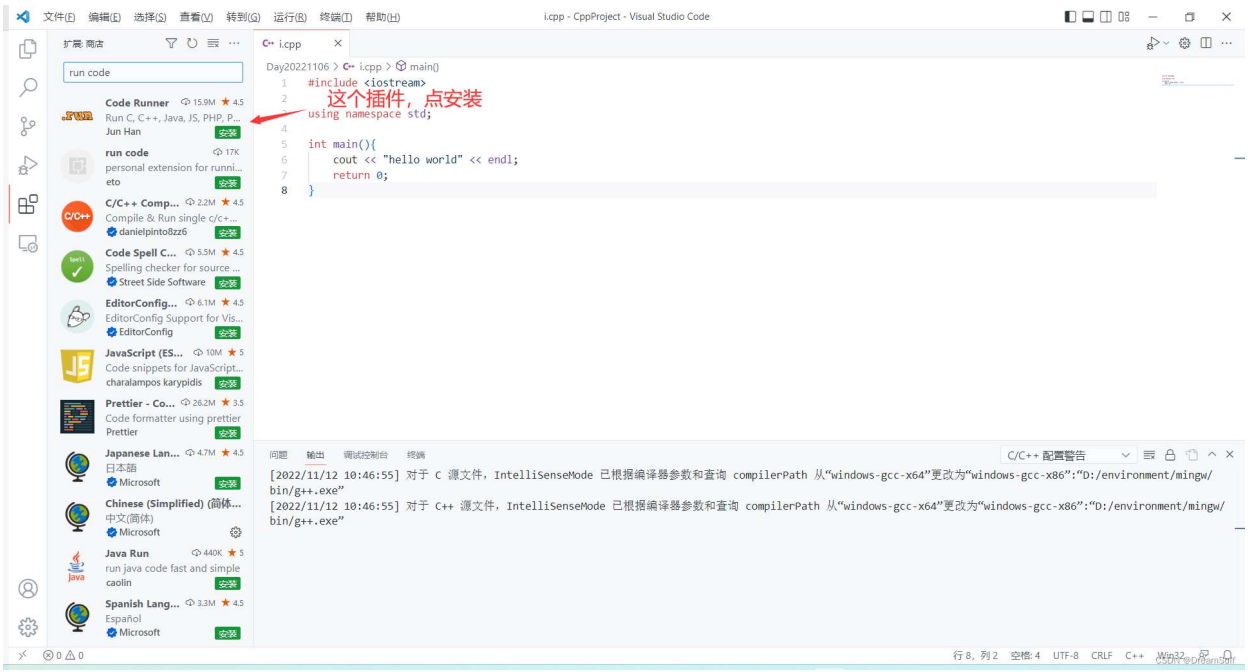




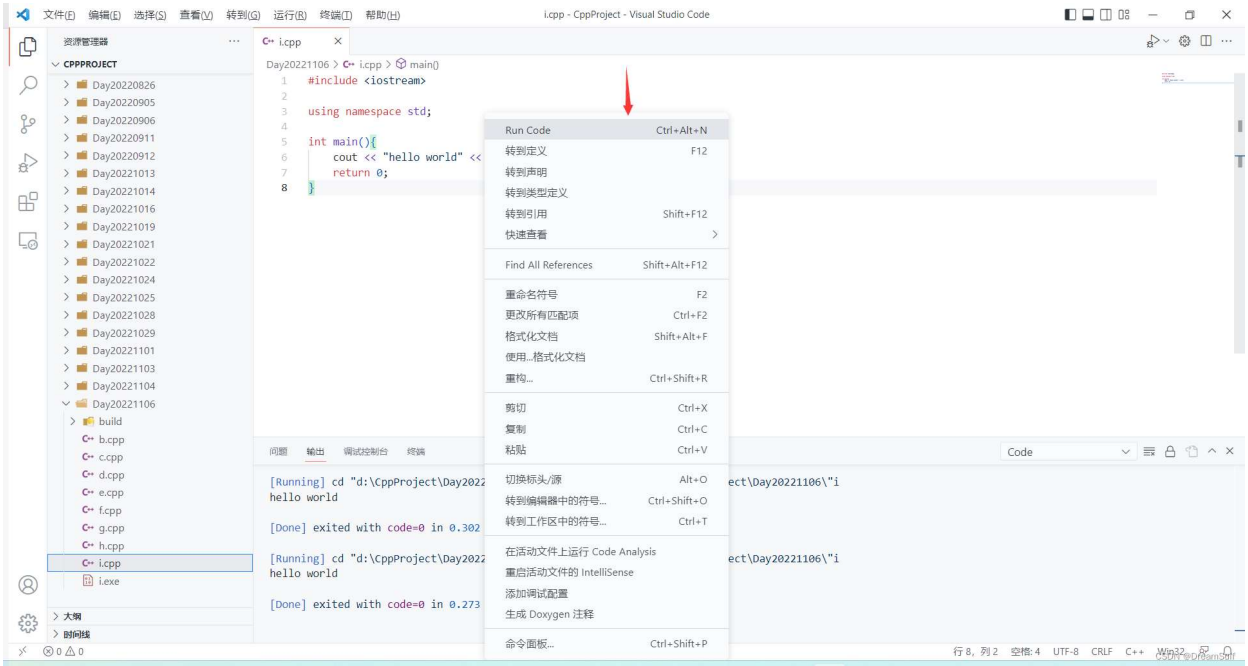
> 然后我们就可以开始配置环境了，配置的环境只适用于这个大文件夹内

第二步 （1）简单的环境配置方法 - run code调试

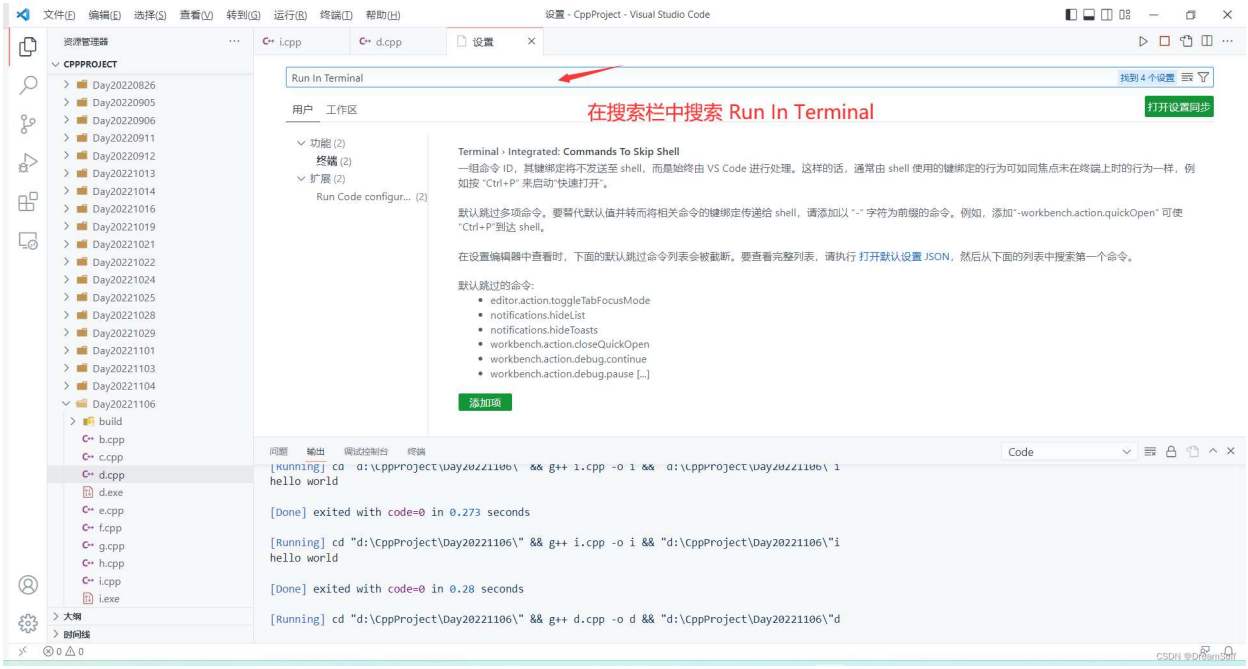
这个方法需要用到一个插件， run code，我们打开扩展商店安装一下run code插件

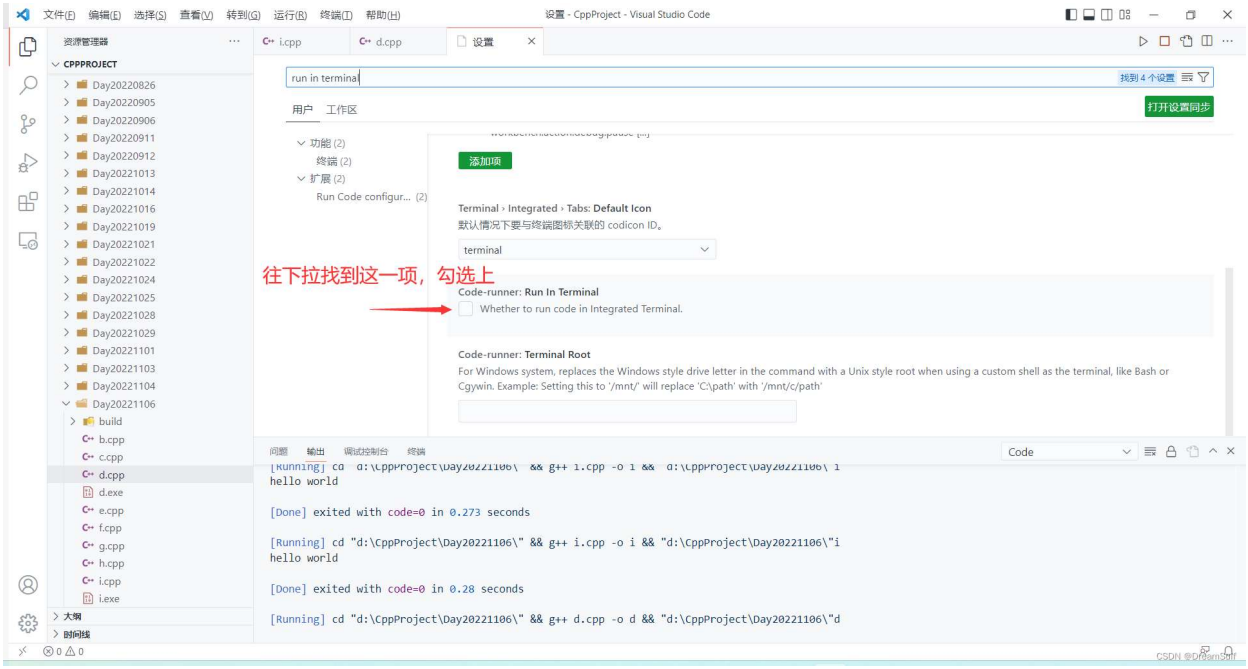


安装完成了后，我们会发现右键多了个run code



直接点run code就能够编译运行程序了，当然，这样只能得到输出结果，而不能直接在vscode中向程序中输入内容，所以我们还需要一些设置来调整一下run code，我们按 ctrl + ，进入设置（，是一个按键，在M键旁边）





重启一下vscode，这样我们就可以在vscode中向程序输入内容了

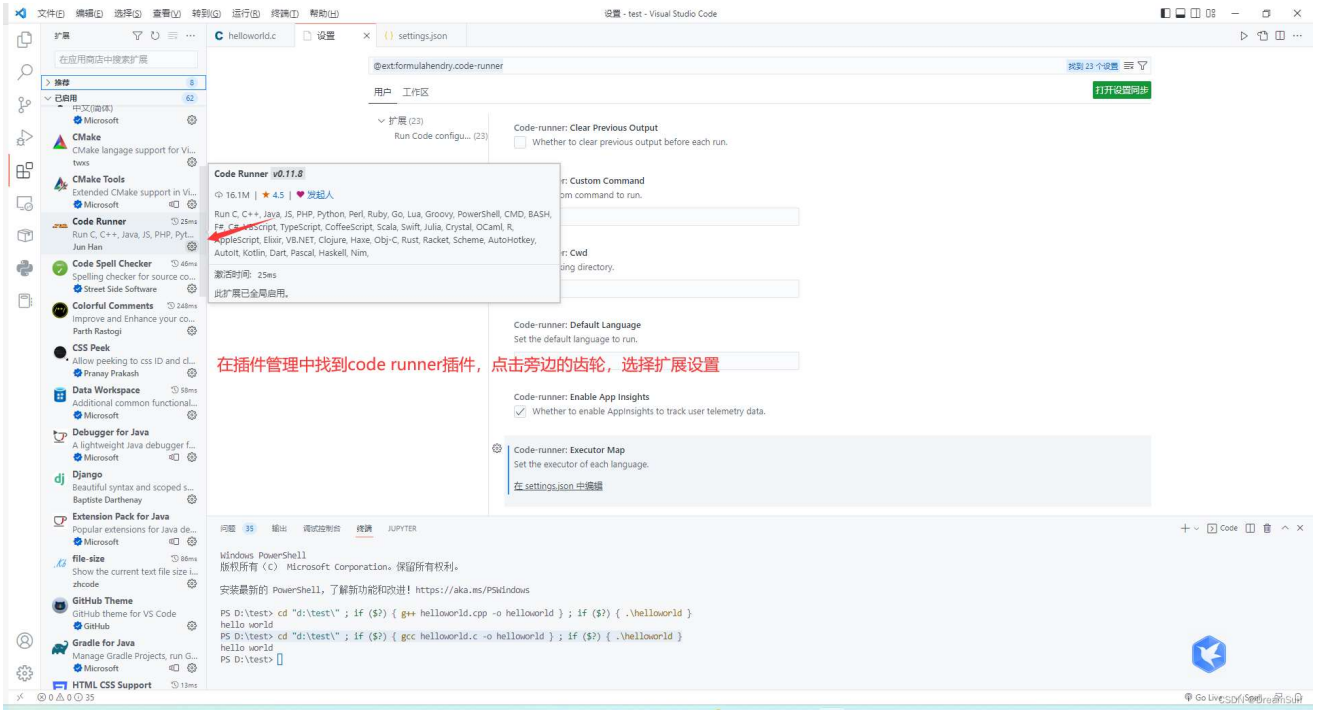
问题1：run code执行代码出现gcc(或g++) :error; no such file or directory的错误

原因分析

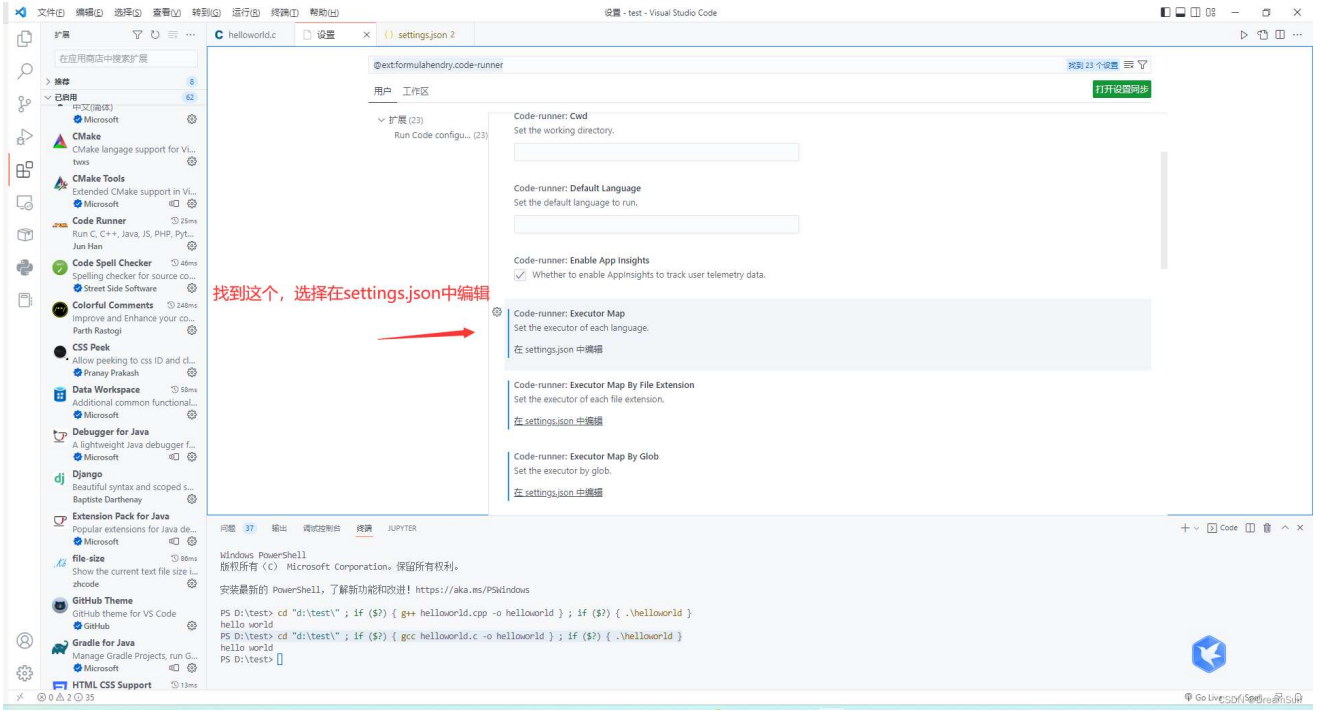
这个错误可能是编译命令执行前没有进入指定的文件夹前导致的，所以我们加一个cd 目标文件夹路径即可

解决办法

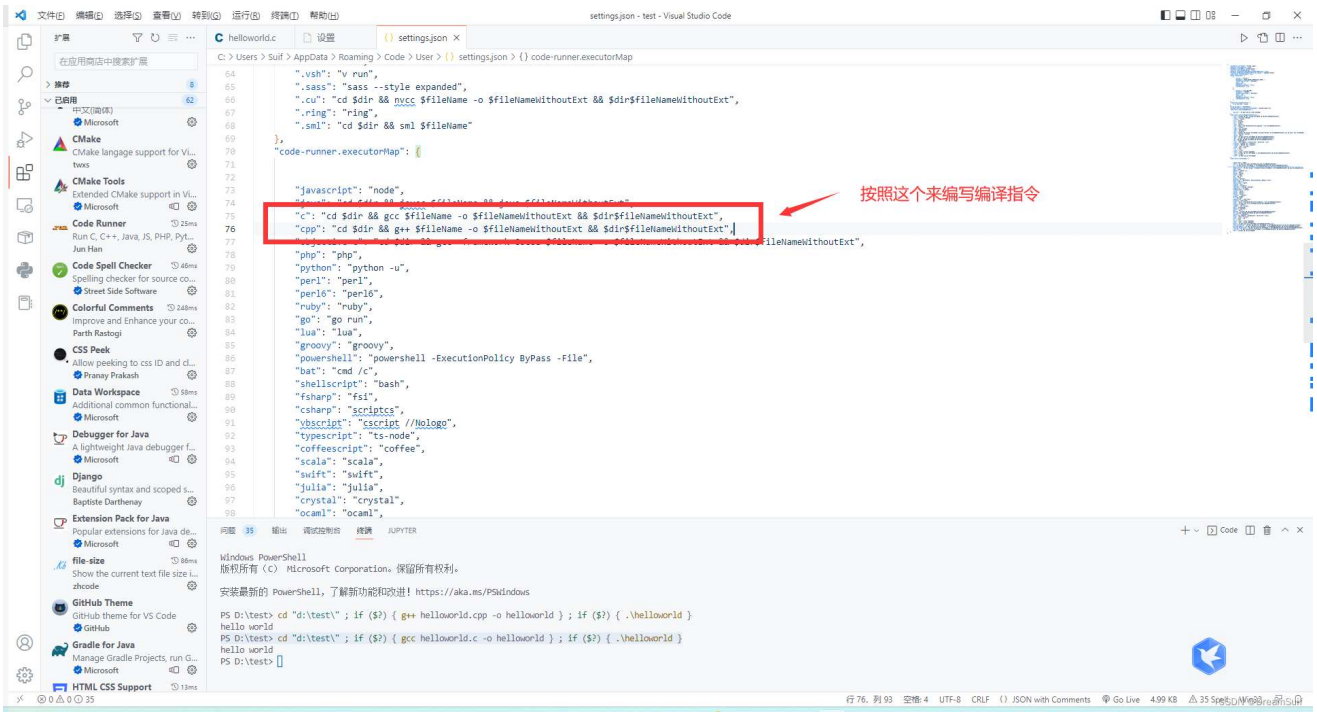
第一步 打开code runner扩展设置



第二步 进入指令设置界面



第三步 修改编译指令




```
        "c": "cd $dir && gcc $fileName -o $fileNameWithoutExt && $dir$fileNameWithoutExt",
        "cpp": "cd $dir && g++ $fileName -o $fileNameWithoutExt && $dir$fileNameWithoutExt",
```

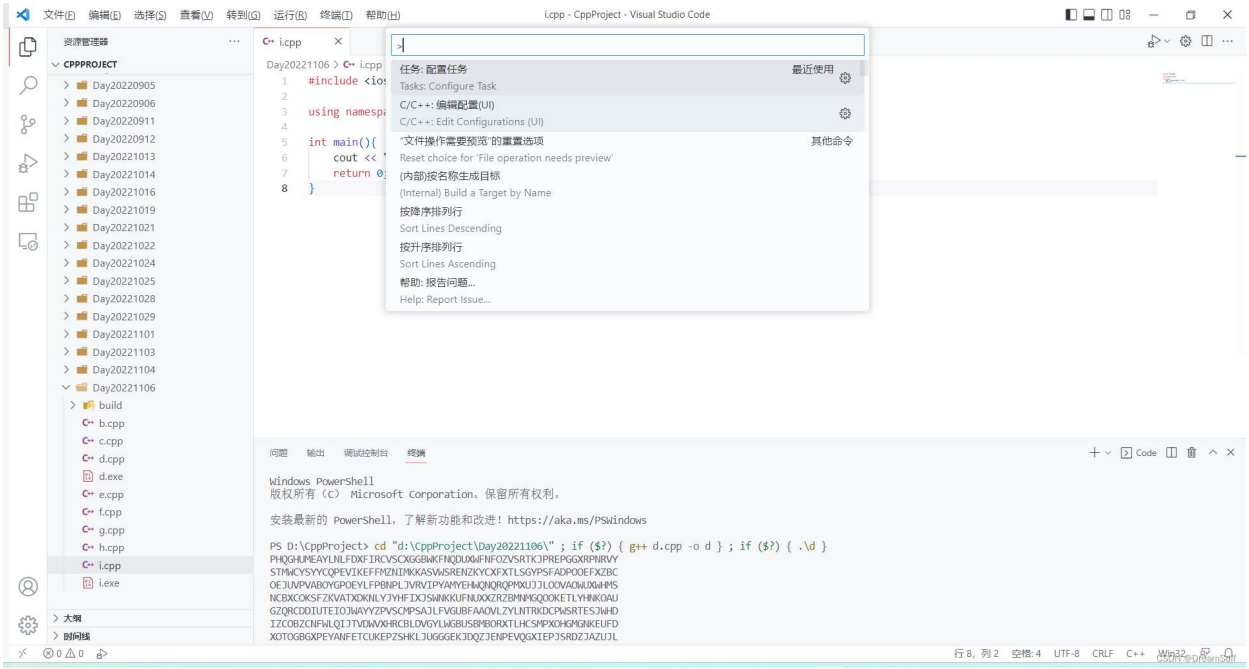
- 1
- 2

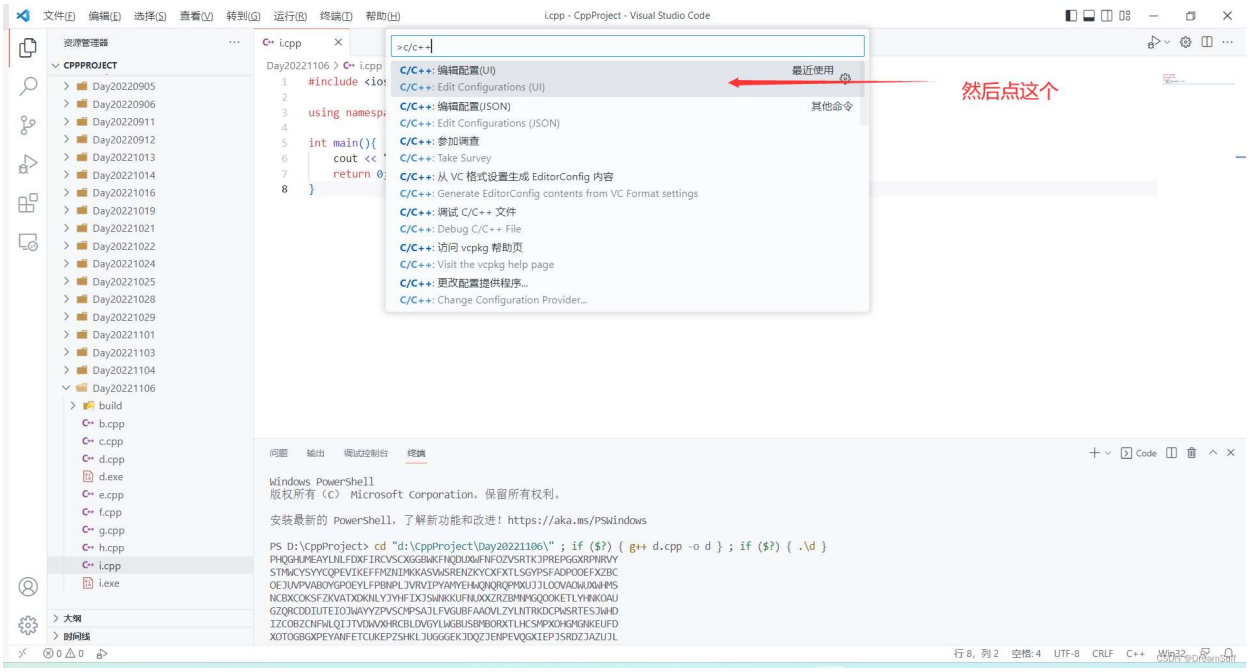
第二步 （2）复杂的环境配置方法 - gdb调试

> 当然，不借助run code的方法我们也有，而且，我个人更推荐这种环境配置方法。

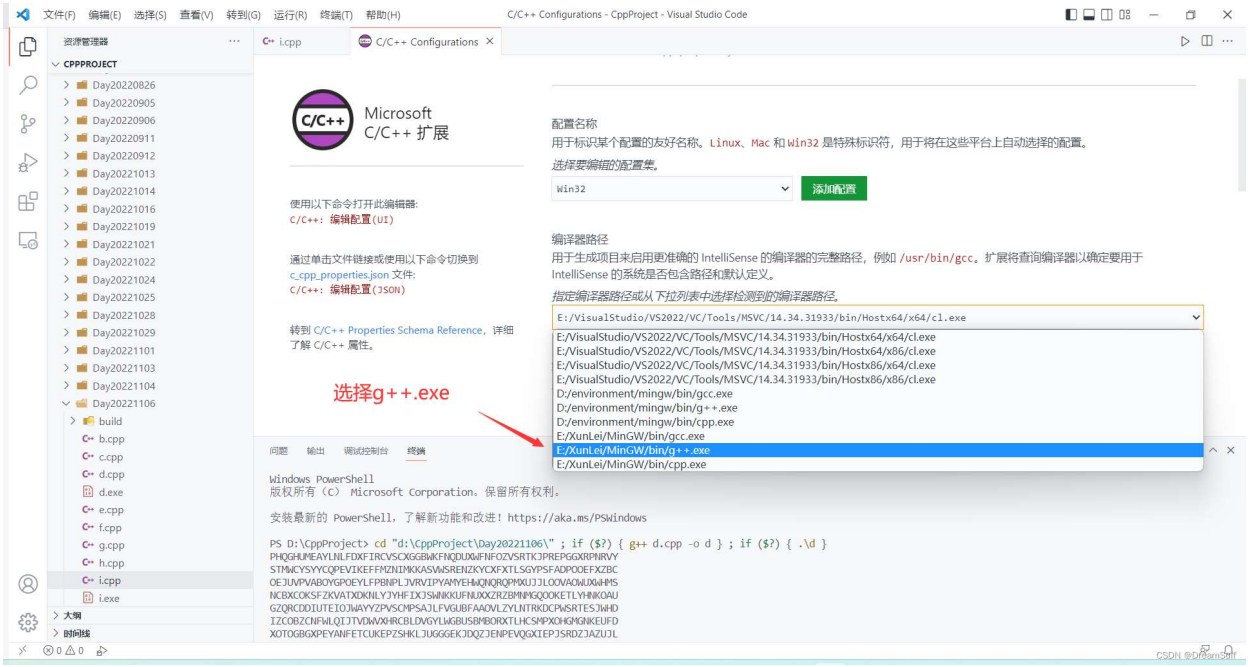
第一步 配置编译器环境

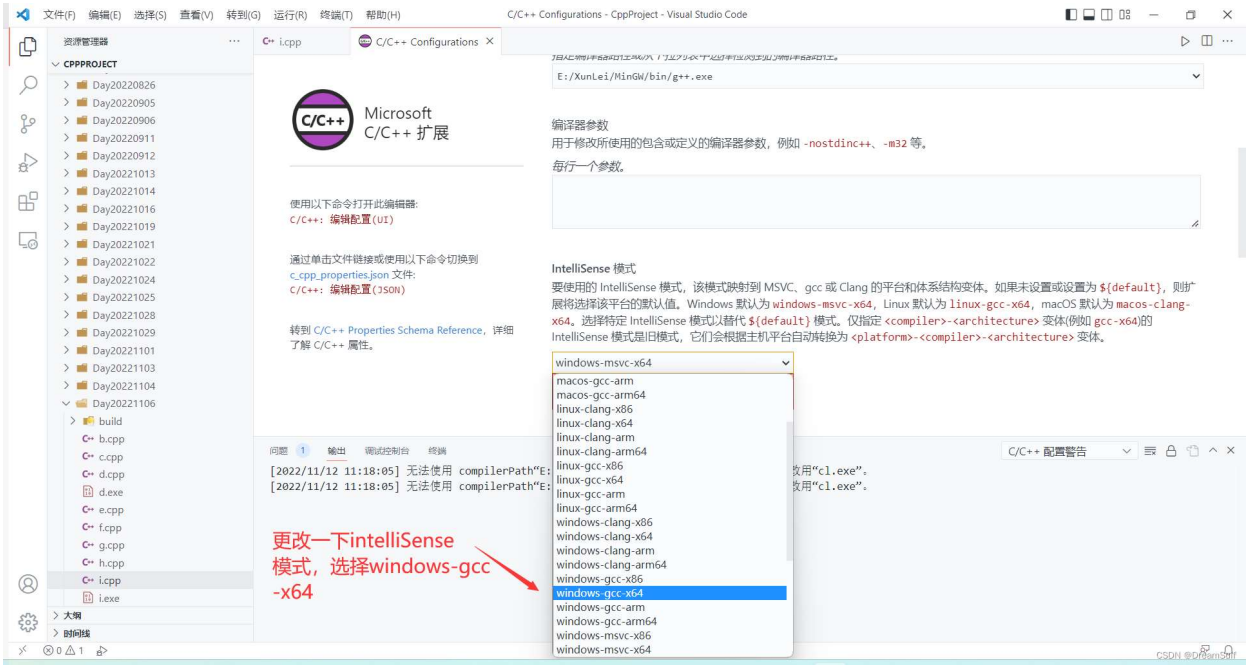
我们按住 ctrl + shift + p



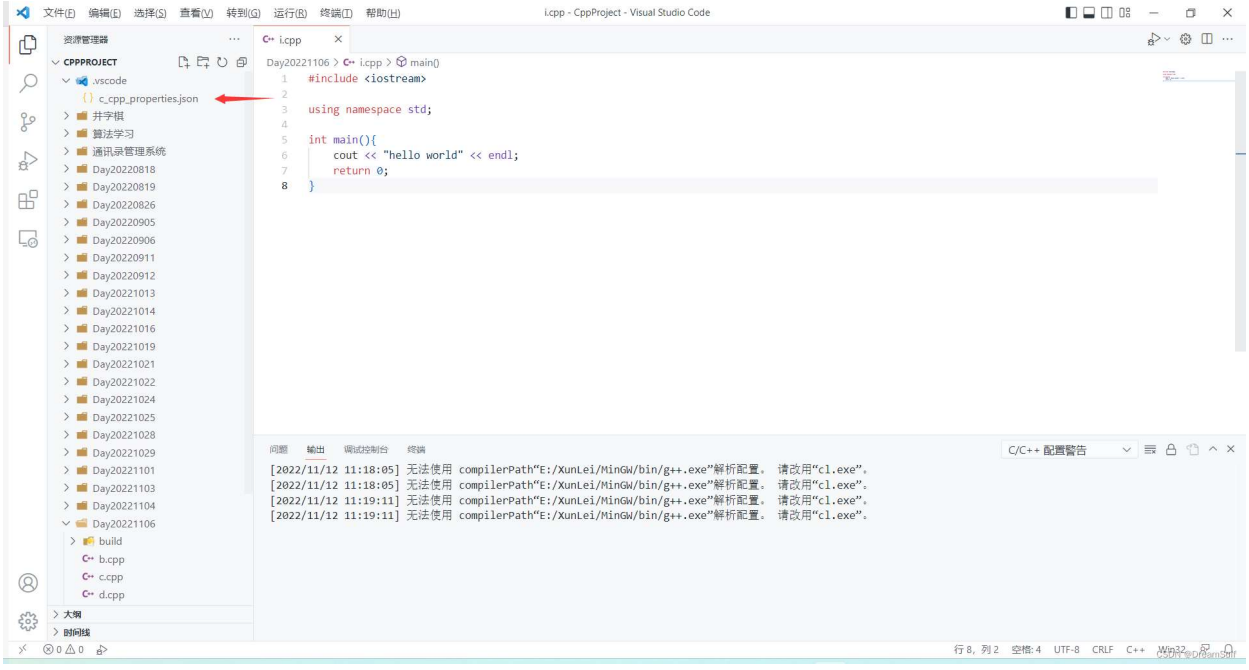


我们配置一下编译器路径



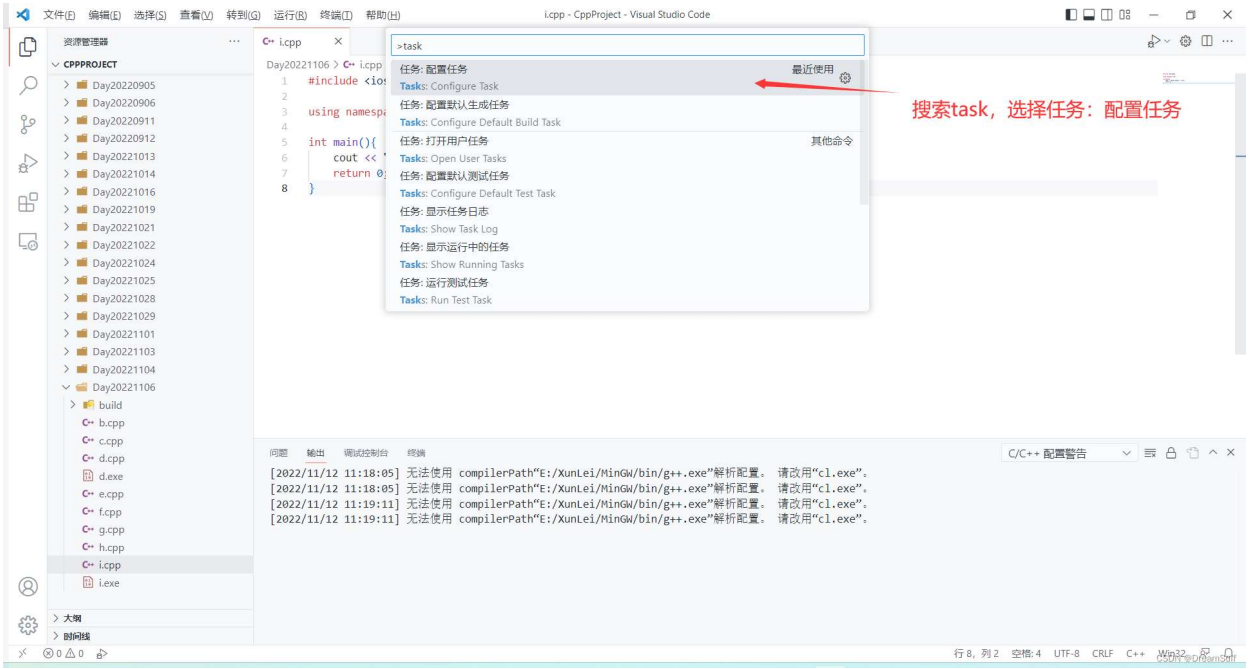


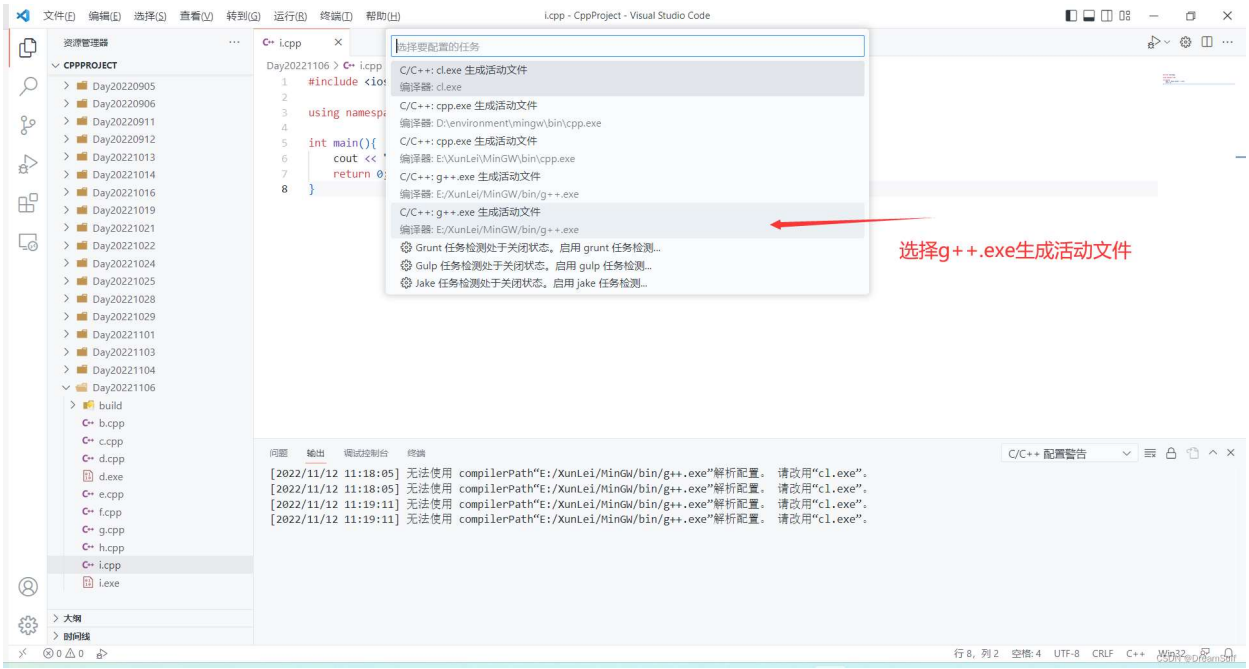
然后关闭这个界面，我们会发现CppProject文件夹中多了一个.vscode文件夹



第二步 配置task编译任务

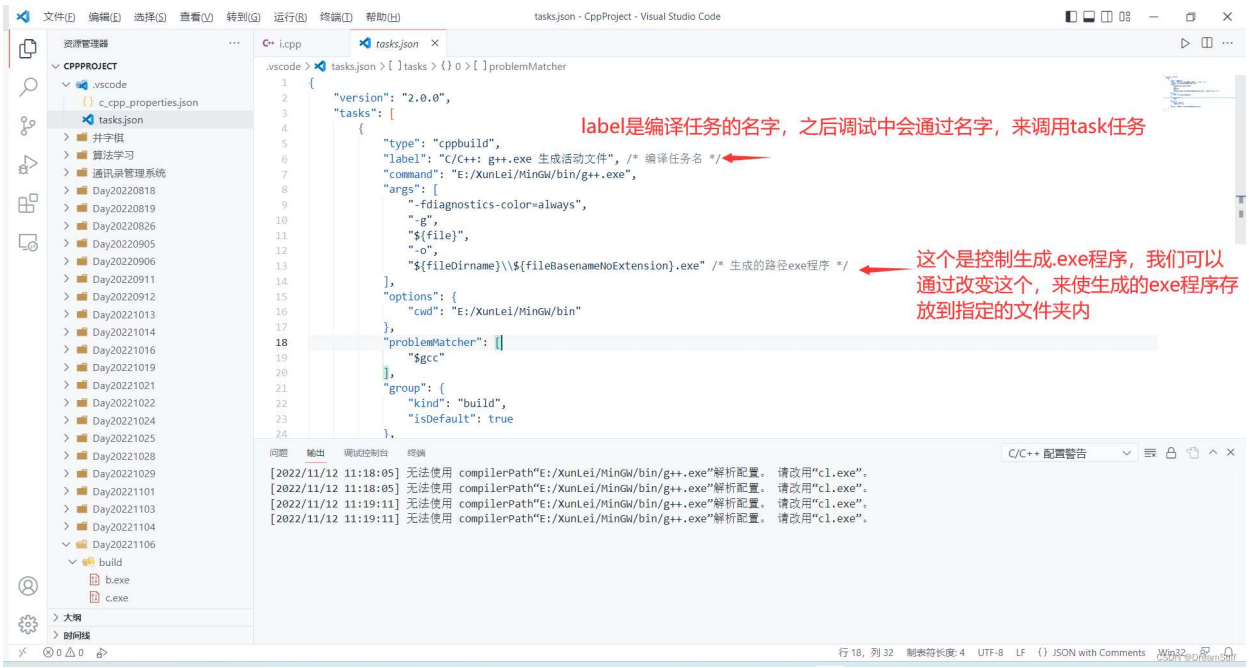
同样，我们按住 ctrl + shift + p





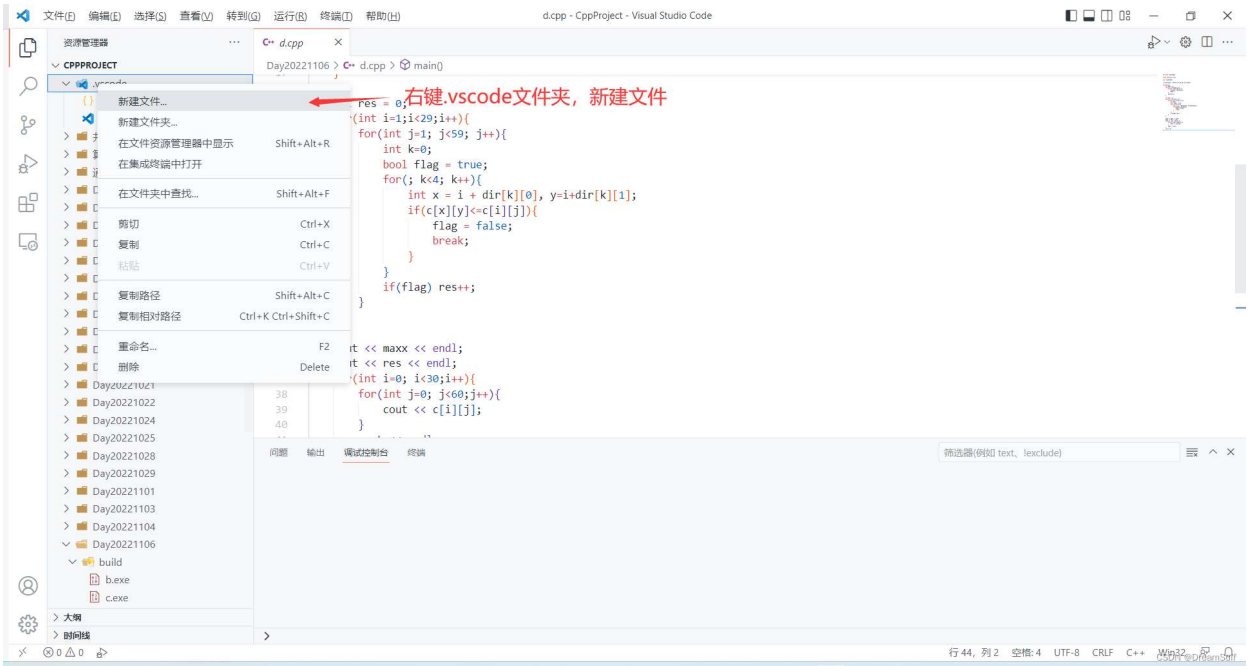
此时我们会发现.vscode文件夹中多了tasks.json

简单地介绍一下tasks.json中需要注意的内容

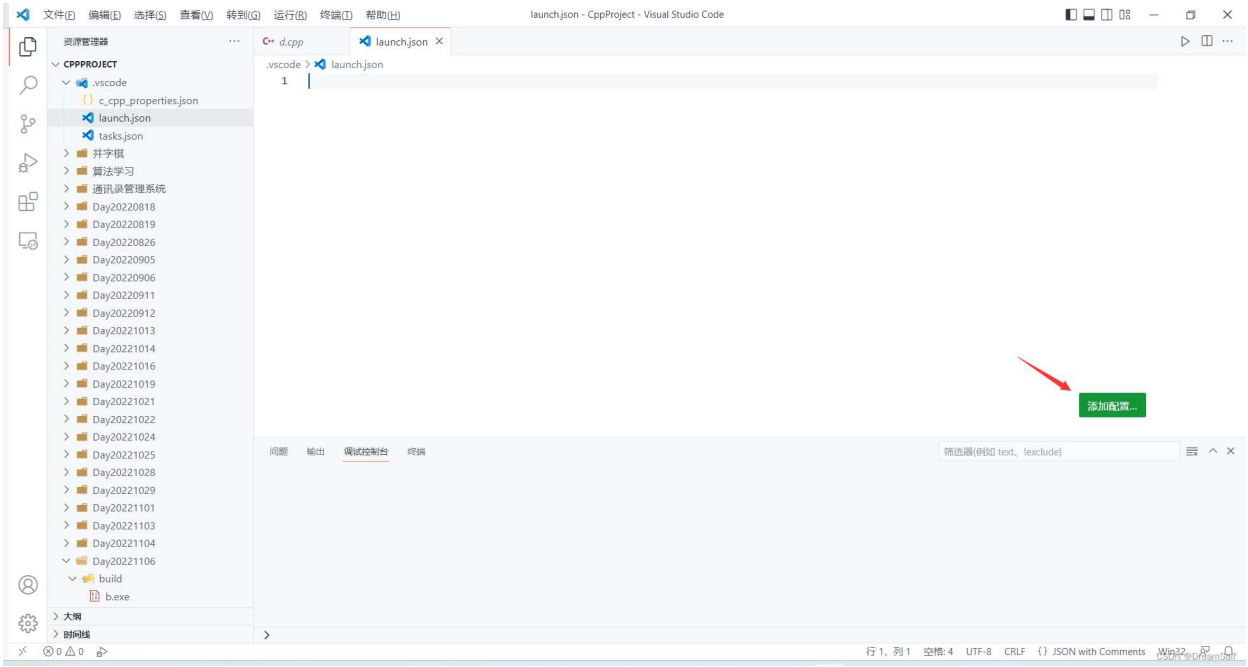


第三步 配置launch调试任务

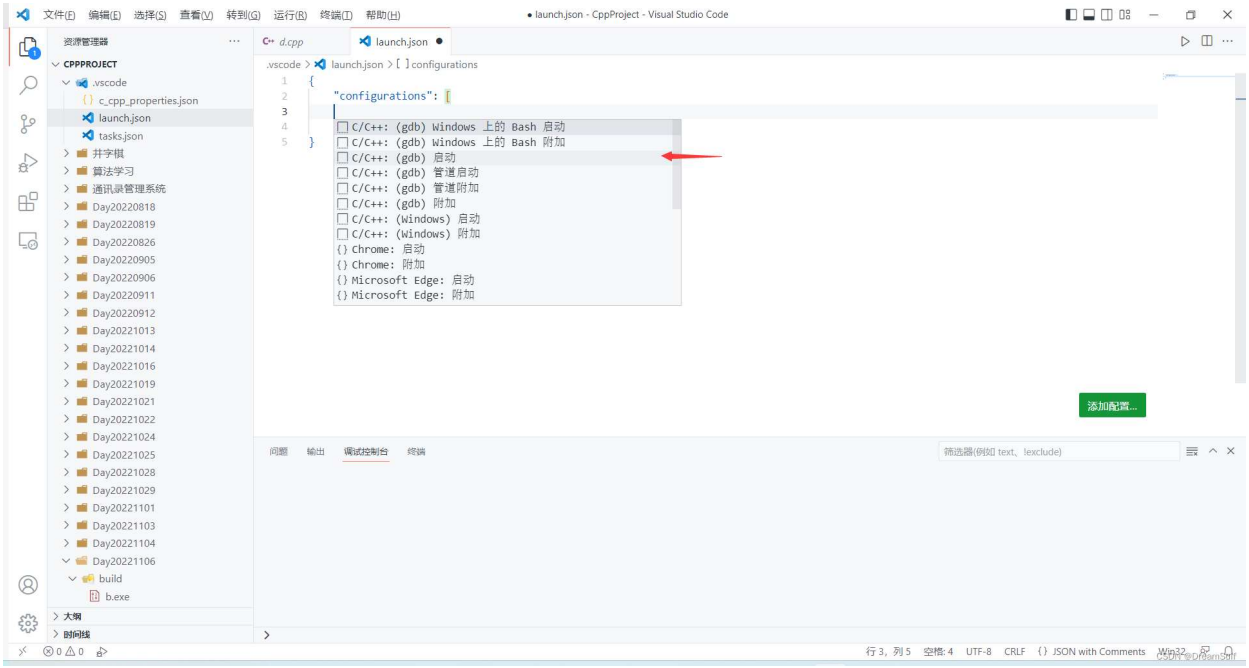
因为C/C++插件不会自动生成launch.json，因此launch.json需要我们自己编写，在.vscode文件夹内新建一个launch.json文件



我们会发现，这里有个添加配置

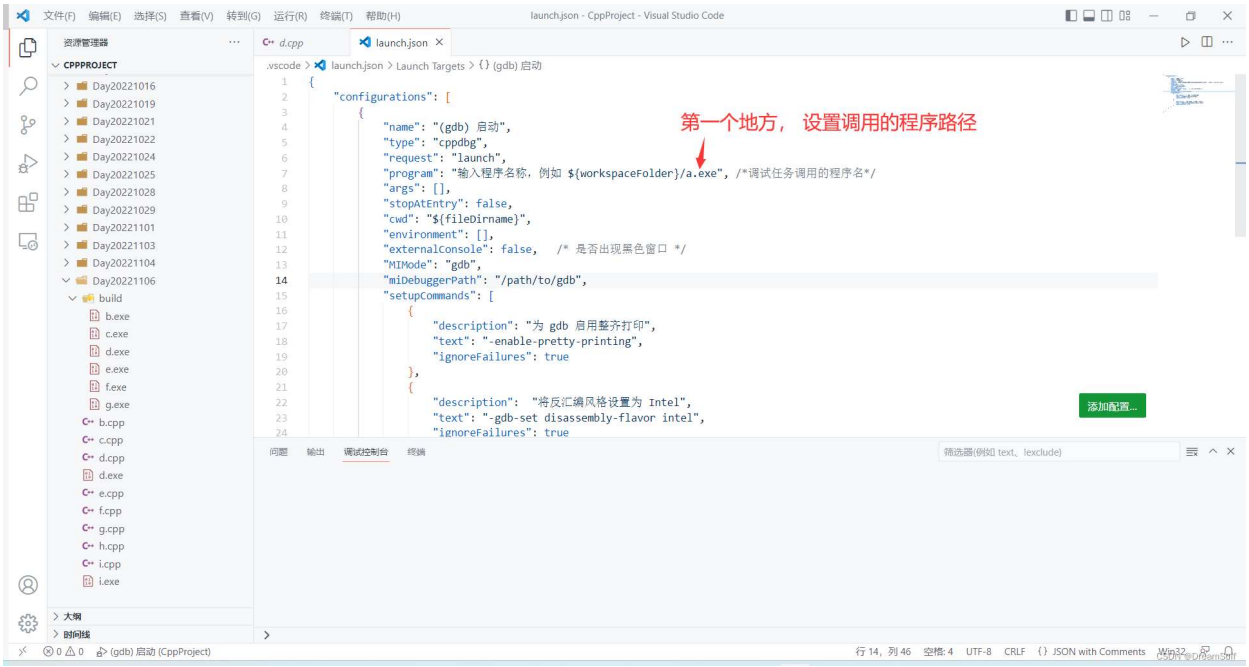


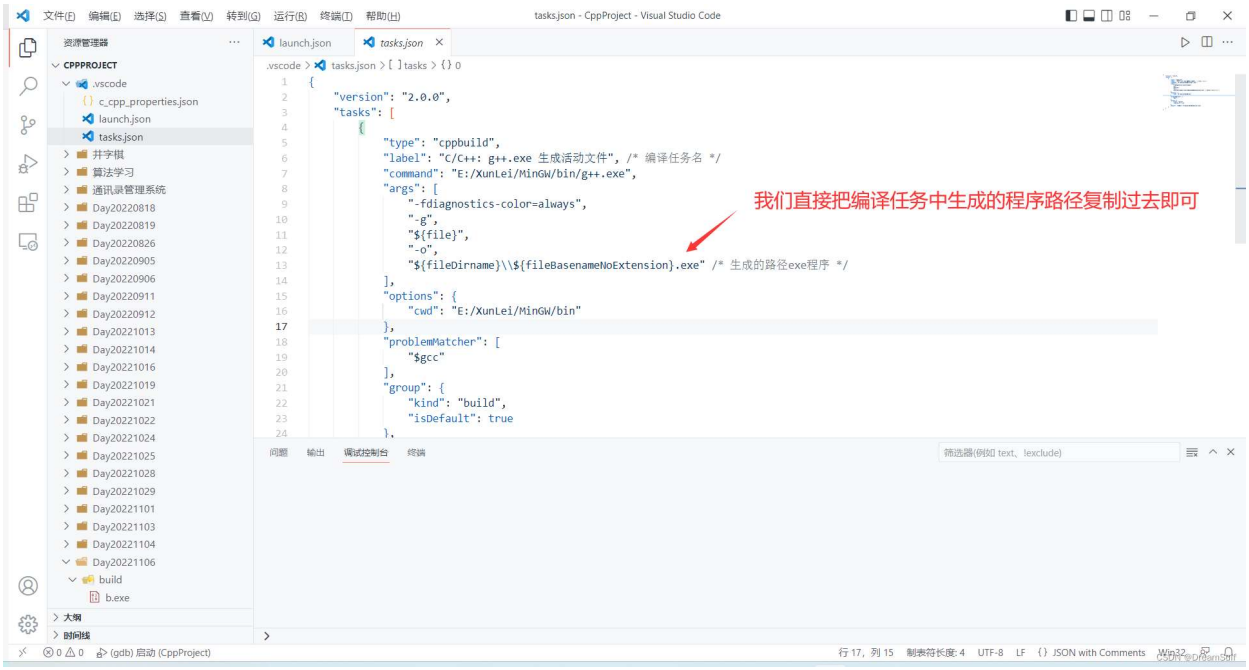
我们点添加配置，选择gdb启动

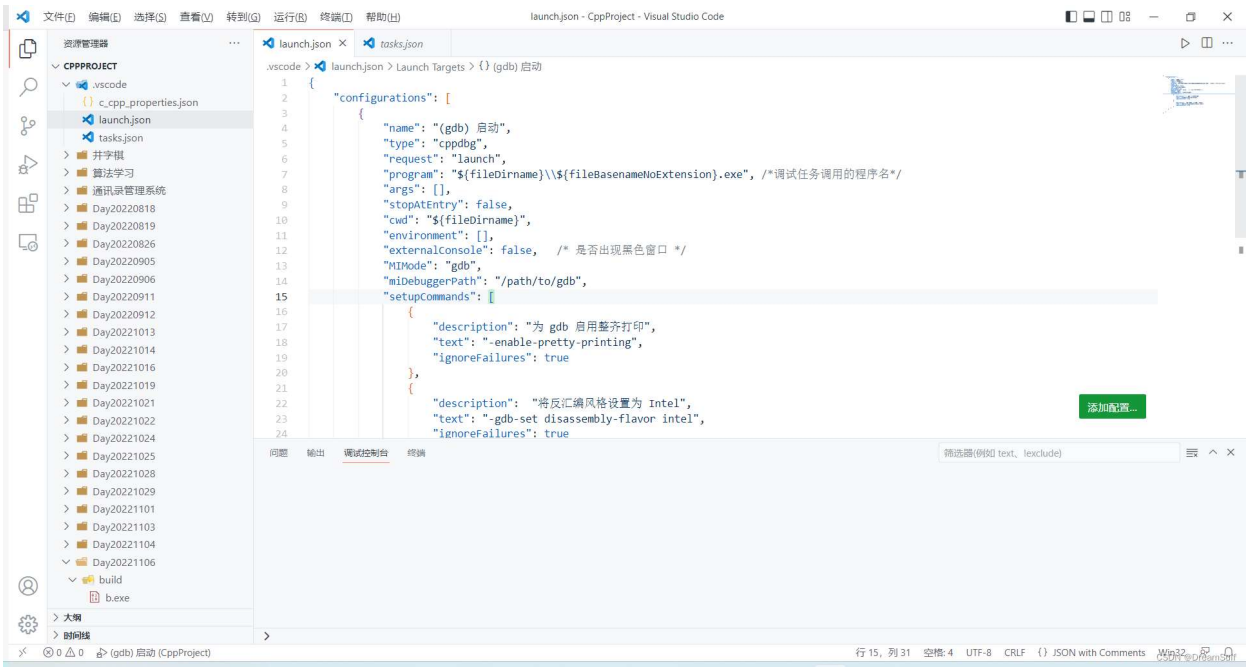


我们需要修改3个地方

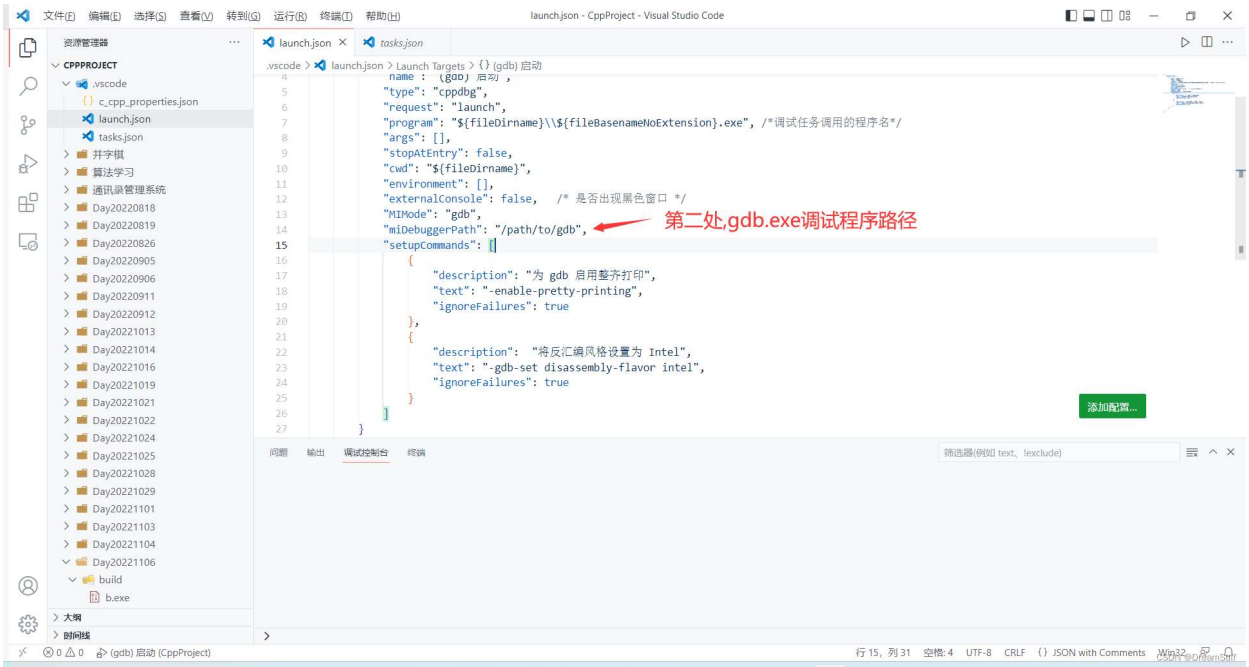
第一处

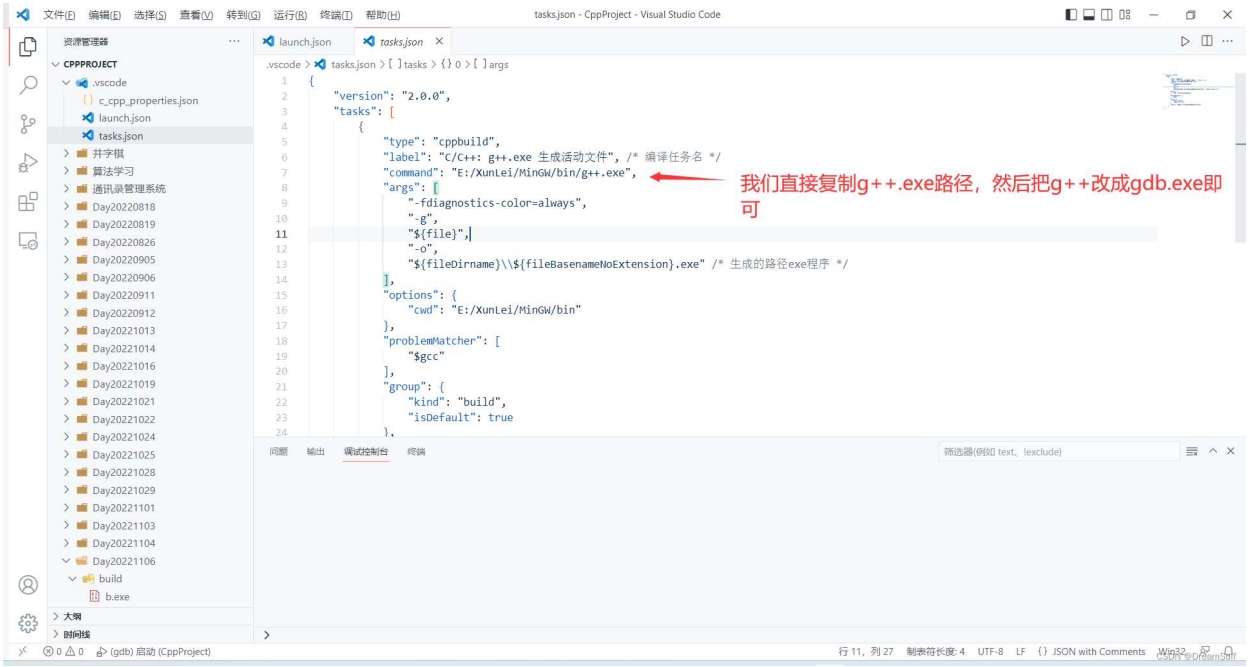

launch.json - CppProject - Visual Studio Code
第一个地方， 设置调用的程序路径
"program": "输入程序名称，例如 ${workspaceFolder}/a.exe", /*调试任务调用的程序名*/
添加配置...

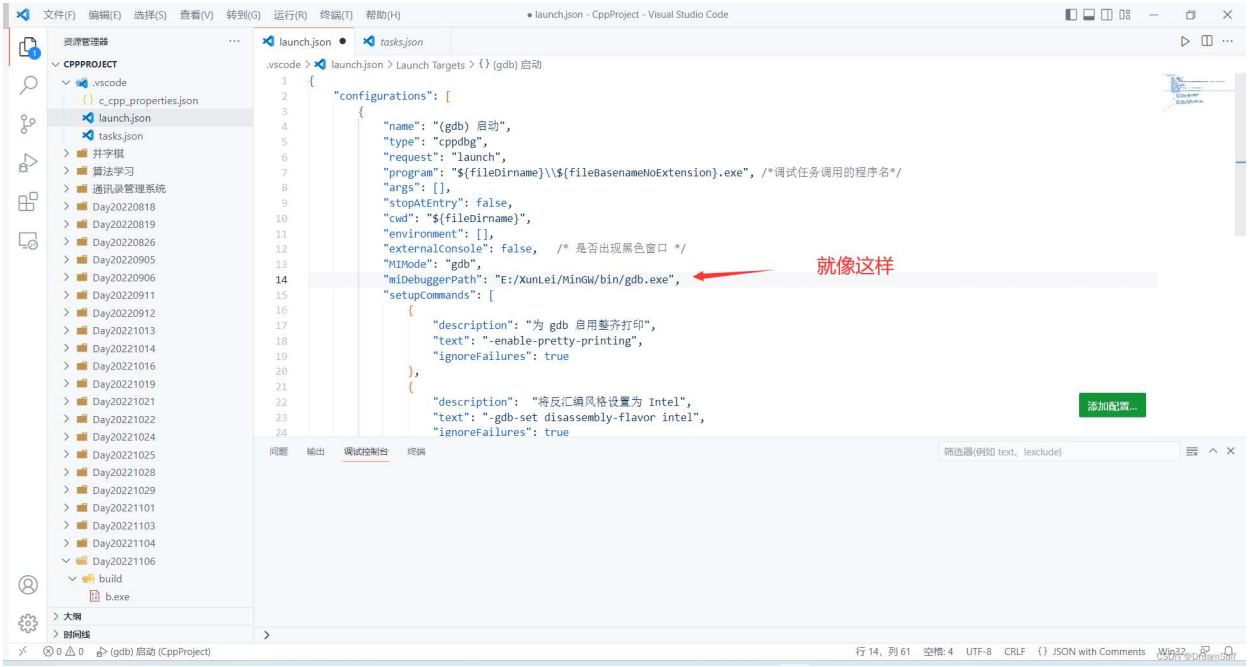

tasks.json - CppProject - Visual Studio Code
我们直接把编译任务中生成的程序路径复制过去即可
"${fileDirname}\\${fileBasenameNoExtension}.exe" /* 生成的路径exe程序 */


launch.json - CppProject - Visual Studio Code
launch.json
tasks.json
.vscode > launch.json > Launch Targets > {} (gdb) 启动
{
"configurations": [
{
"name": "(gdb) 启动",
"type": "cppdbg",
"request": "launch",
"program": "${fileDirname}\\${fileBasenameNoExtension}.exe", /*调试任务调用的程序名*/
"args": [],
"stopAtEntry": false,
"cwd": "${fileDirname}",
"environment": [],
"externalConsole": false, /* 是否出现黑色窗口 */
"MIMode": "gdb",
"miDebuggerPath": "/path/to/gdb",
"setupCommands": [
{
"description": "为 gdb 启用整齐打印",
"text": "-enable-pretty-printing",
"ignoreFailures": true
},
{
"description": "将反汇编风格设置为 Intel",
"text": "-gdb-set disassembly-flavor intel",
"ignoreFailures": true
添加配置...
问题
输出
调试控制台
终端
筛选器(例如 text、!exclude)
(gdb) 启动 (CppProject)
行 15，列 31
空格: 4
UTF-8
CRLF
JSON with Comments

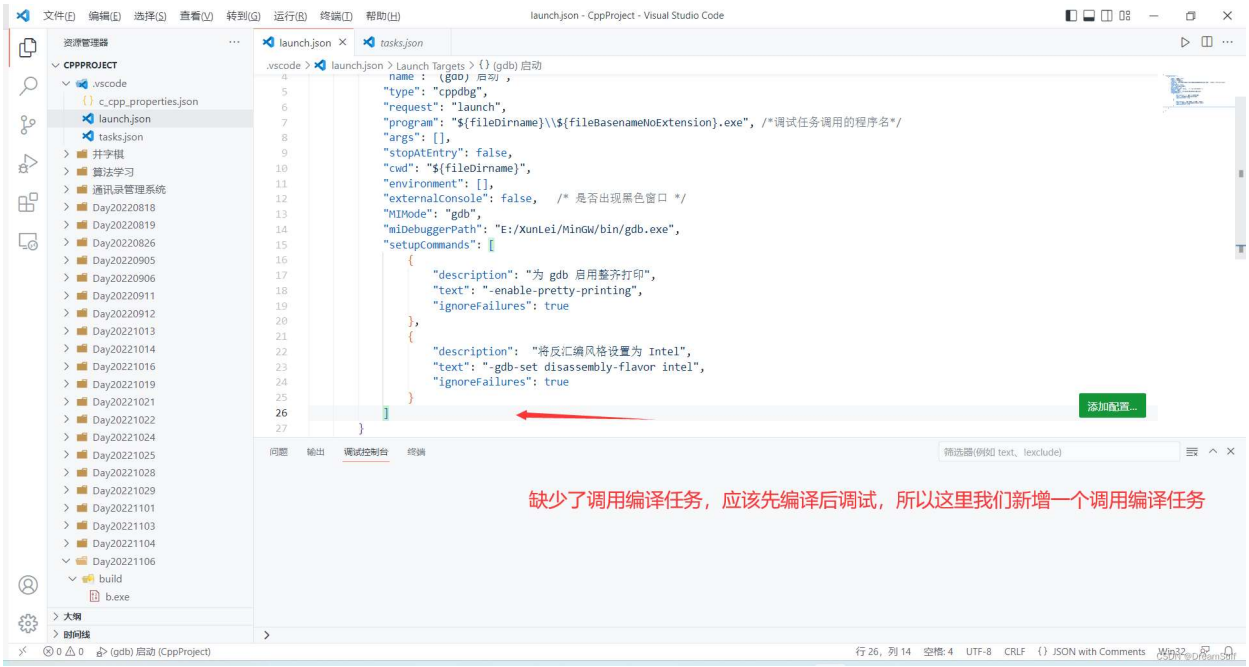
第二处

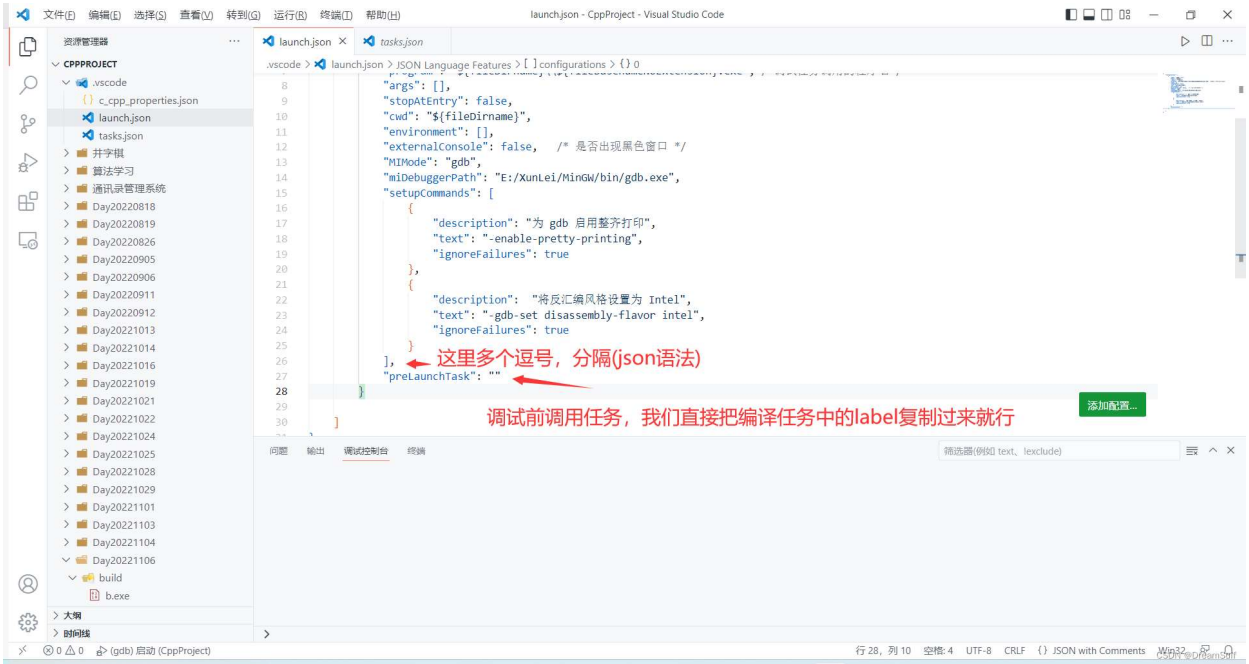

launch.json - CppProject - Visual Studio Code
launch.json
tasks.json
.vscode > launch.json > Launch Targets > {} (gdb) 启动
"type": "cppdbg",
"request": "launch",
"program": "${fileDirname}\\${fileBasenameNoExtension}.exe", /*调试任务调用的程序名*/
"args": [],
"stopAtEntry": false,
"cwd": "${fileDirname}",
"environment": [],
"externalConsole": false, /* 是否出现黑色窗口 */
"MIMode": "gdb",
"miDebuggerPath": "/path/to/gdb",
第二处,gdb.exe调试程序路径
"setupCommands": [
{
"description": "为 gdb 启用整齐打印",
"text": "-enable-pretty-printing",
"ignoreFailures": true
},
{
"description": "将反汇编风格设置为 Intel",
"text": "-gdb-set disassembly-flavor intel",
"ignoreFailures": true
}
]
}
添加配置...
问题
输出
调试控制台
终端
筛选器(例如 text、!exclude)
(gdb) 启动 (CppProject)
行 15，列 31
空格: 4
UTF-8
CRLF
JSON with Comments

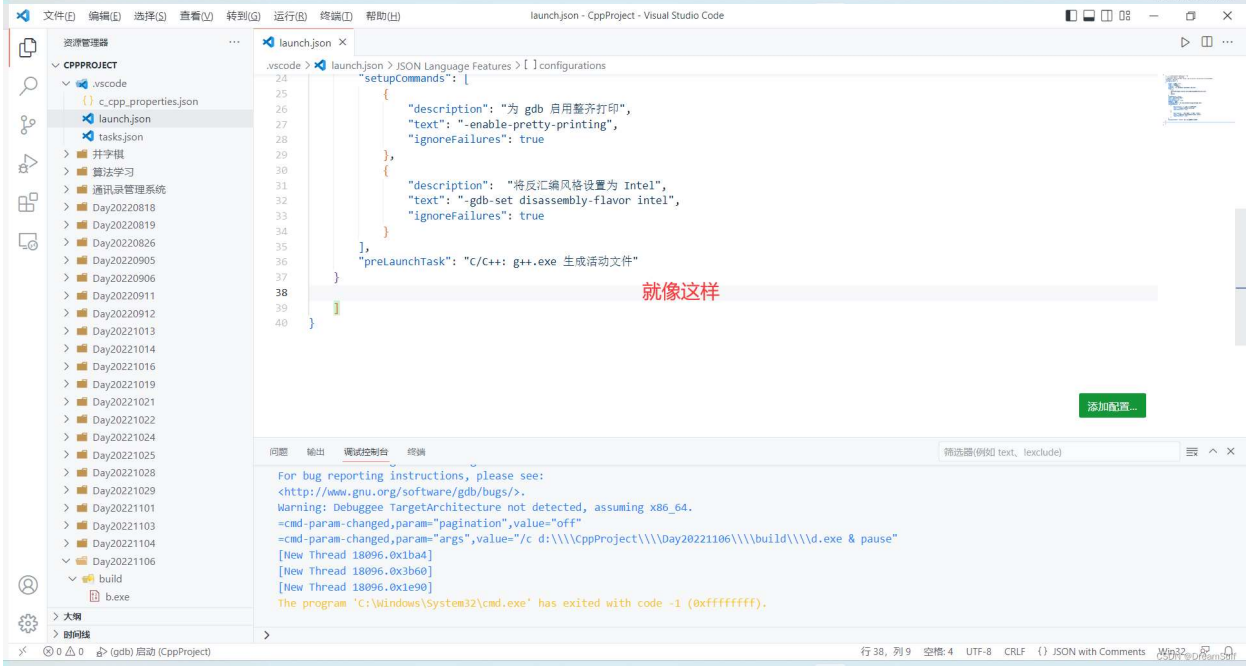

tasks.json - CppProject - Visual Studio Code
launch.json
tasks.json
.vscode > tasks.json > [ ] tasks > {} 0 > [ ] args
{
"version": "2.0.0",
"tasks": [
{
"type": "cppbuild",
"label": "C/C++: g++.exe 生成活动文件", /* 编译任务名 */
"command": "E:/XunLei/MinGW/bin/g++.exe",
"args": [
"-fdiagnostics-color=always",
"-g",
"${file}",
"-o",
"${fileDirname}\\${fileBasenameNoExtension}.exe" /* 生成的路径exe程序 */
],
"options": {
"cwd": "E:/XunLei/MinGW/bin"
},
"problemMatcher": [
"$gcc"
],
"group": {
"kind": "build",
"isDefault": true
},
我们直接复制g++.exe路径，然后把g++改成gdb.exe即可
CPPPROJECT
.vscode
c_cpp_properties.json
launch.json
tasks.json
build
b.exe
问题
输出
调试控制台
终端
筛选器(例如 text、!exclude)
(gdb) 启动 (CppProject)
行 11，列 27
制表符长度: 4
UTF-8
LF
JSON with Comments


launch.json - CppProject - Visual Studio Code
launch.json
tasks.json
.vscode > launch.json > Launch Targets > {} (gdb) 启动
{
"configurations": [
{
"name": "(gdb) 启动",
"type": "cppdbg",
"request": "launch",
"program": "${fileDirname}\\${fileBasenameNoExtension}.exe", /*调试任务调用的程序名*/
"args": [],
"stopAtEntry": false,
"cwd": "${fileDirname}",
"environment": [],
"externalConsole": false, /* 是否出现黑色窗口 */
"MIMode": "gdb",
"miDebuggerPath": "E:/XunLei/MinGW/bin/gdb.exe",
"setupCommands": [
{
"description": "为 gdb 启用整齐打印",
"text": "-enable-pretty-printing",
"ignoreFailures": true
},
{
"description": "将反汇编风格设置为 Intel",
"text": "-gdb-set disassembly-flavor intel",
"ignoreFailures": true
就像这样
添加配置...
CPPPROJECT
.vscode
c_cpp_properties.json
launch.json
tasks.json
build
b.exe
问题
输出
调试控制台
终端
筛选器(例如 text、!exclude)
(gdb) 启动 (CppProject)
行 14，列 61
空格: 4
UTF-8
CRLF
JSON with Comments

第三处


launch.json - CppProject - Visual Studio Code
launch.json
tasks.json
.vscode > launch.json > Launch Targets > {} (gdb) 启动
"type": "cppdbg",
"request": "launch",
"program": "${fileDirname}\\${fileBasenameNoExtension}.exe", /*调试任务调用的程序名*/
"args": [],
"stopAtEntry": false,
"cwd": "${fileDirname}",
"environment": [],
"externalConsole": false, /* 是否出现黑色窗口 */
"MIMode": "gdb",
"miDebuggerPath": "E:/XunLei/MinGW/bin/gdb.exe",
"setupCommands": [
{
"description": "为 gdb 启用整齐打印",
"text": "-enable-pretty-printing",
"ignoreFailures": true
},
{
"description": "将反汇编风格设置为 Intel",
"text": "-gdb-set disassembly-flavor intel",
"ignoreFailures": true
}
]
}
添加配置...
缺少了调用编译任务，应该先编译后调试，所以这里我们新增一个调用编译任务





保存，重启一下编辑器，我们就可以按F5调试程序了

注意：任何文件做出更改后都需要保存，新内容才会生效，我们可以按 ctrl + s 快速保存文件

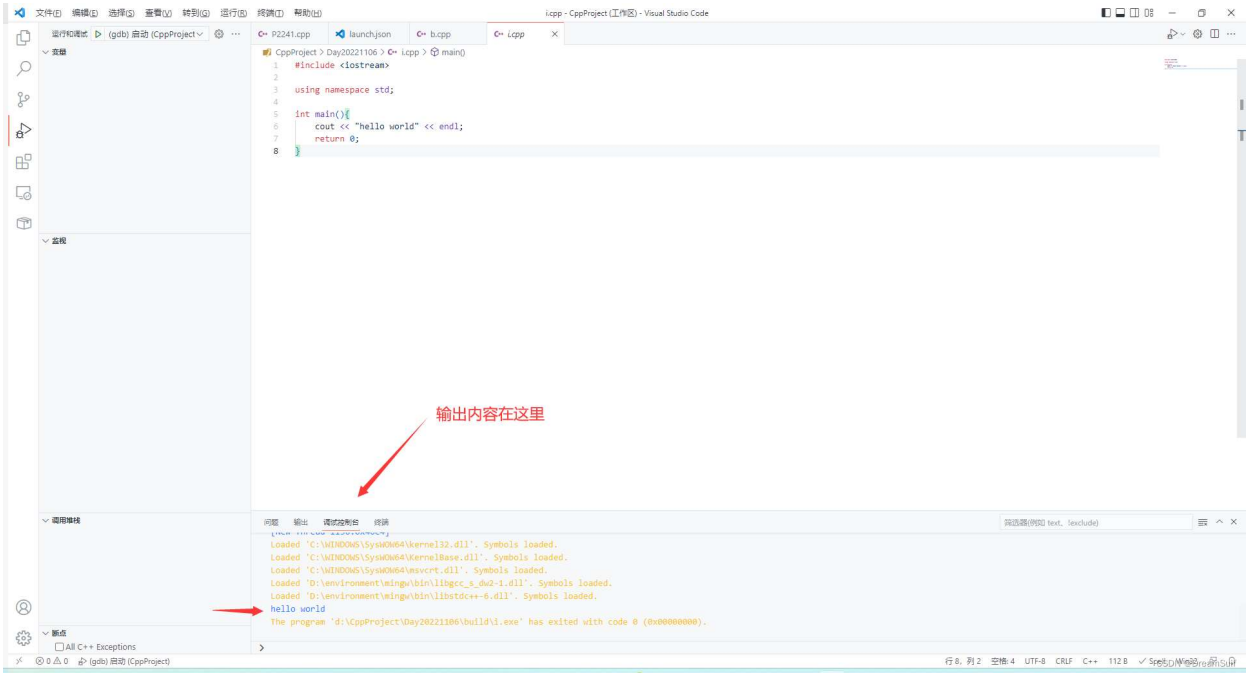
一些问题

问题1

问：为什么按F5后出现这个报错信息，没有输出结果

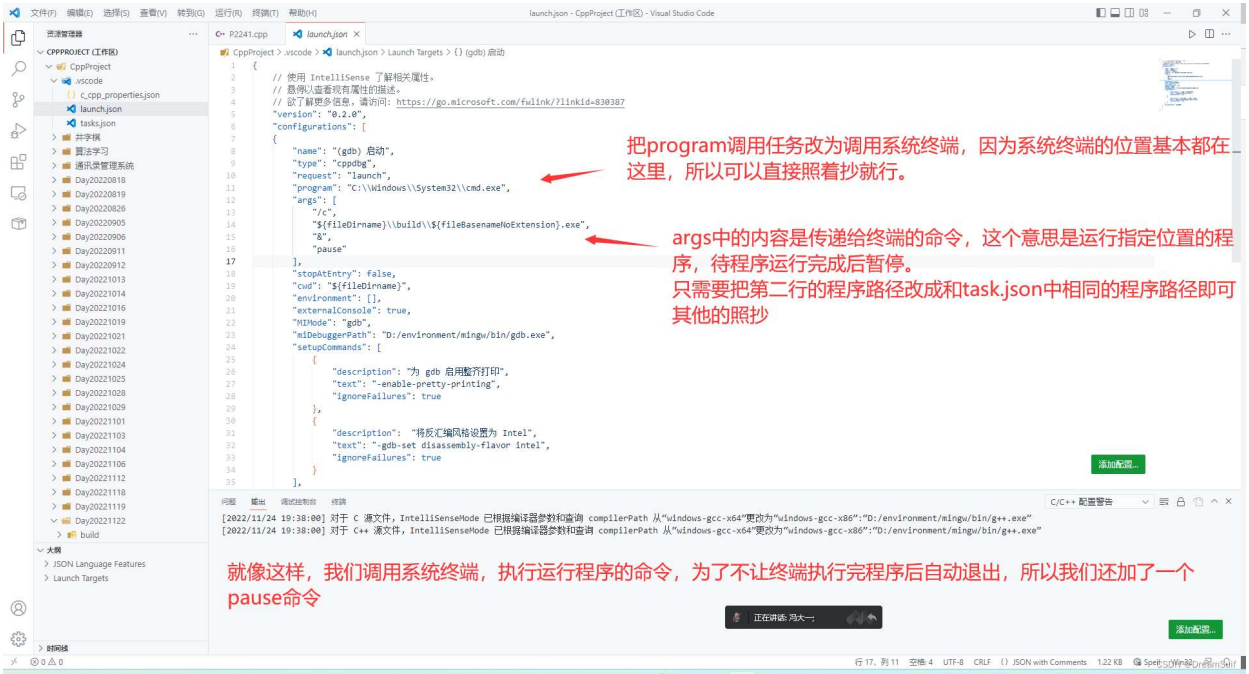
```
PS D:\CppProject> & 'c:\Users\Suif\.vscode\extensions\ms-vscode.cpptools-1.12.4-win32-x64\debugAdapters\bin\WindowsDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-v1llhmtf.3a4' '--stdout=Microsoft-MIEngine-Out-u2ktfyvz.ta2' '--stderr=Microsoft-MIEngine-Error-dbdueza1.v2m' '--pid=Microsoft-MIEngine-Pid-dc0mvv2i.rtr' '--dbgExe=D:/environment/mingw/bin/gdb.exe' '--interpreter=mi'
```

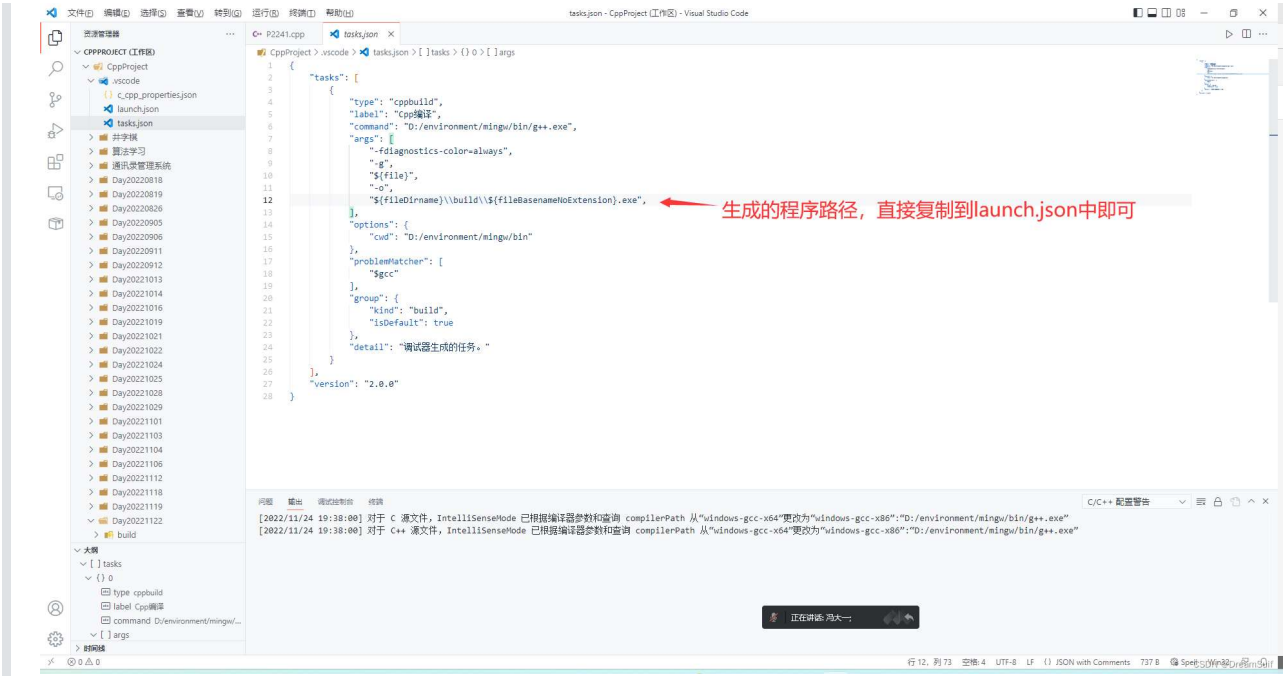
答：这并不是报错信息，而是编译调试命令，输出的结果在调试控制台中显示。

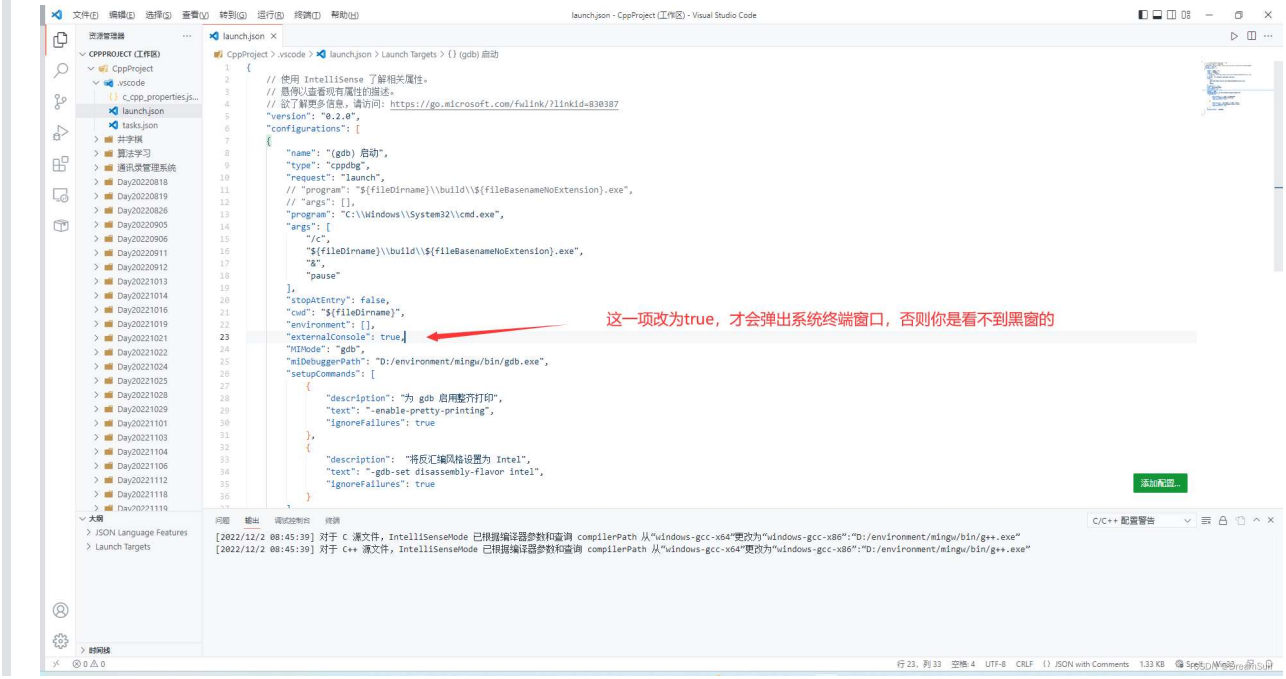


进阶 设置成经典的弹出黑窗运行程序的形式（在系统终端中运行程序）

有的小伙伴可能觉得调试控制台不太好用，想要vscode编译运行c/c++程序时弹出终端，在终端中运行程序，这点其实也很简单，我们只需要改变launch.json中的 program 和 args 中的内容，然后设置一下黑窗弹出即可。





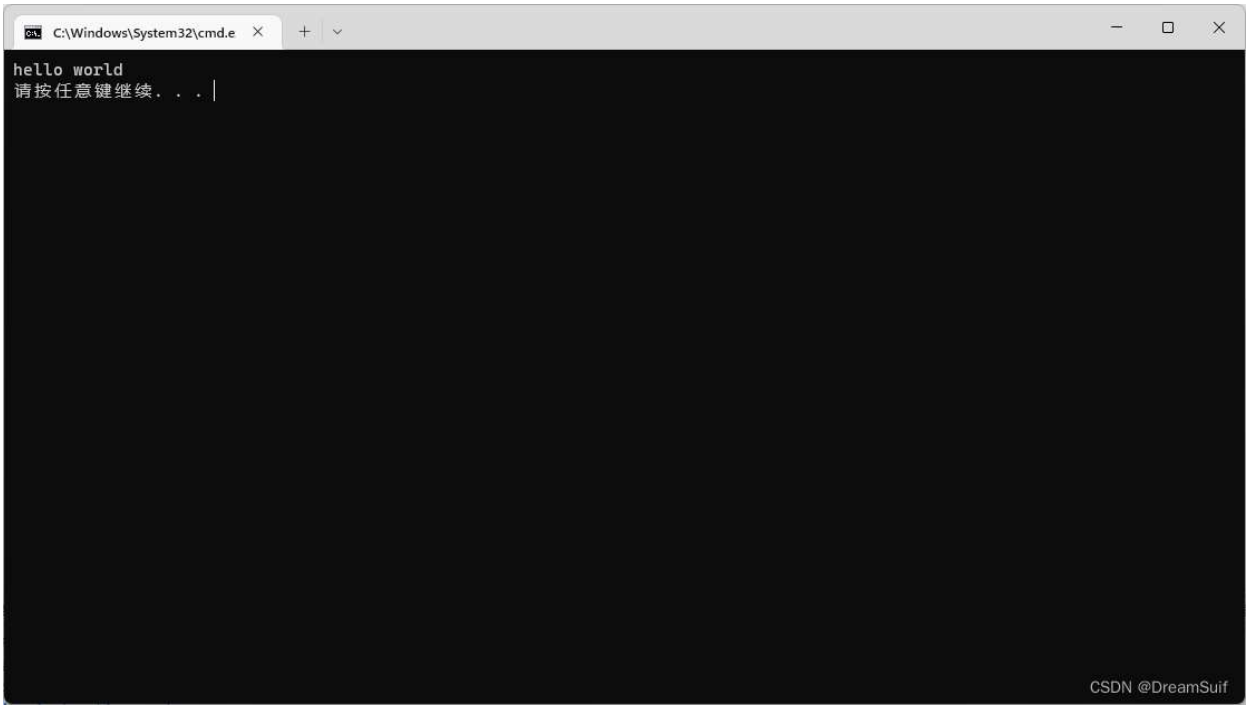


我的program 和 args 中的内容：

```
        "program": "C:\\Windows\\System32\\cmd.exe",
        "args": [
            "/c",
            "${fileDirname}\\build\\${fileBasenameNoExtension}.exe",
            "&",
            "pause"
        ],
```

- 1
- 2
- 3
- 4
- 5
- 6
- 7

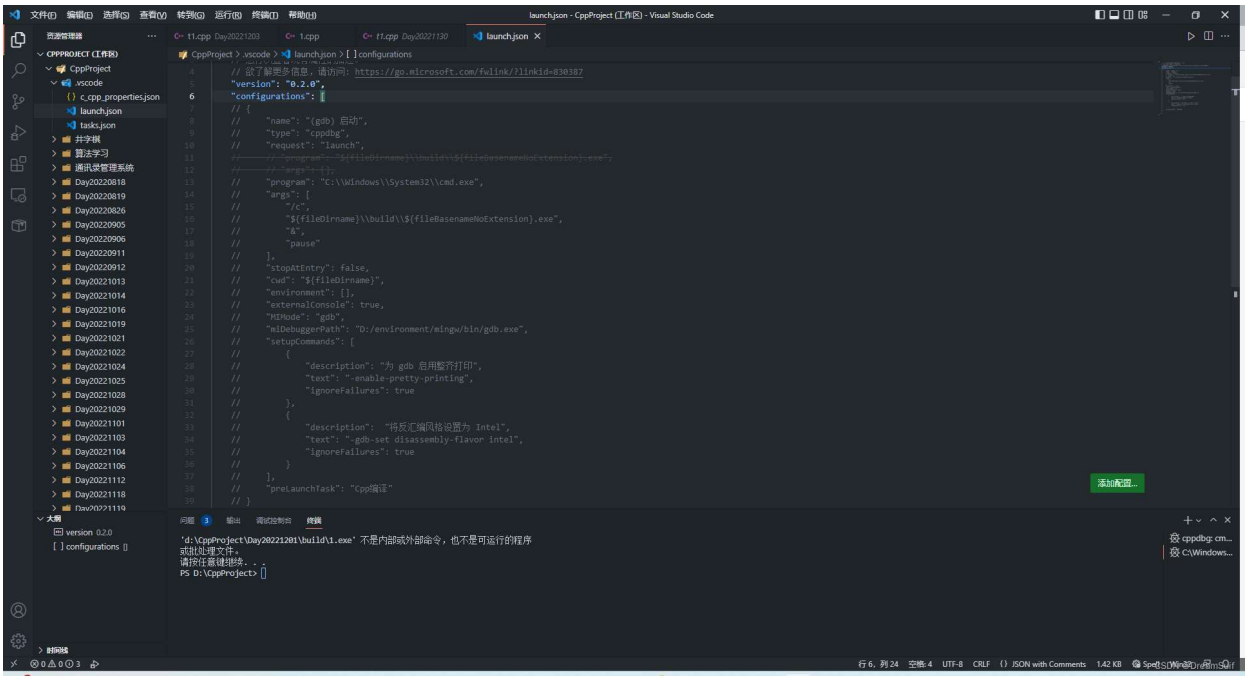
> 完成后的效果



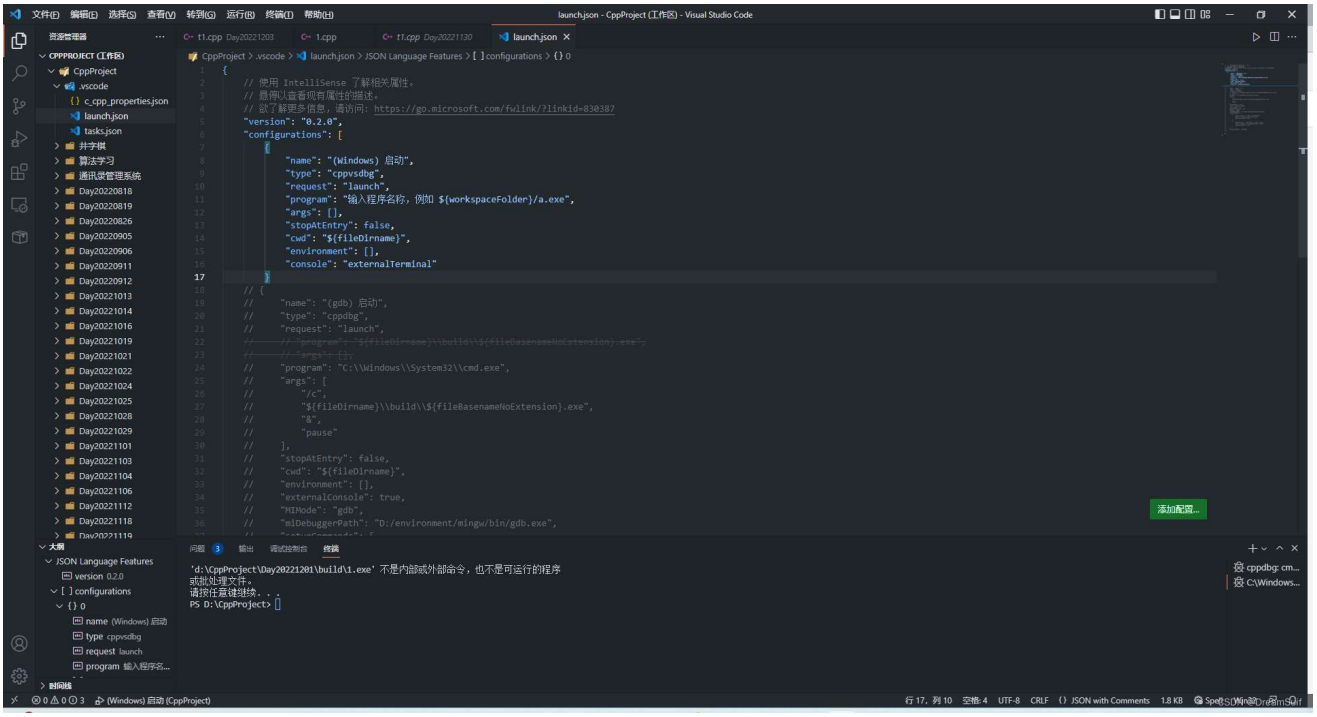
进阶 设置在vscode内置终端中执行程序

> 有的小伙伴既不想用run code等插件，又想要在vscode内置终端中运行程序，这种方法我们当然也有！
> 我们只需要改变launch.json中的内容即可。（因为gdb调试不能在vscode内置终端中执行，所以我们得选择window启动调试任务）
> 先把configurations中的内容注释掉（因为后面配置过程中需要用到之前的一些内容），全选

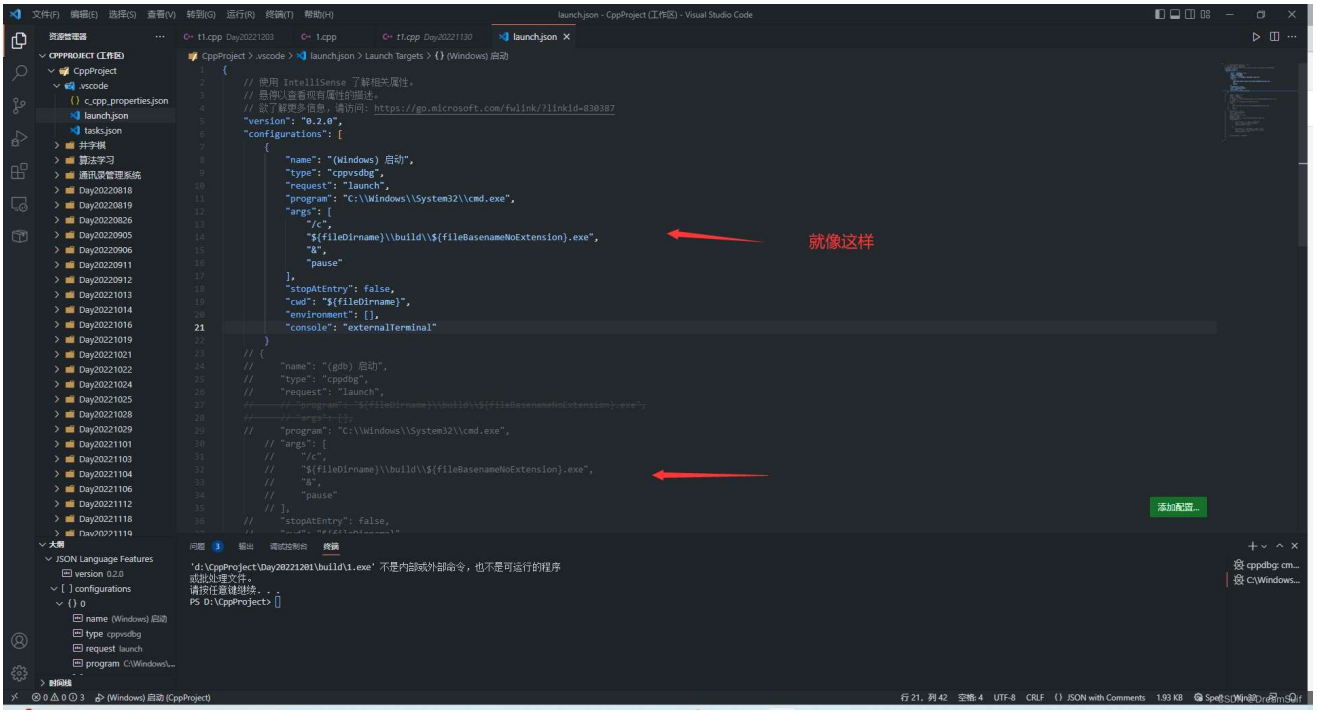
configurations中的内容，按下快捷键ctrl + / 即可快速注释选中内容，



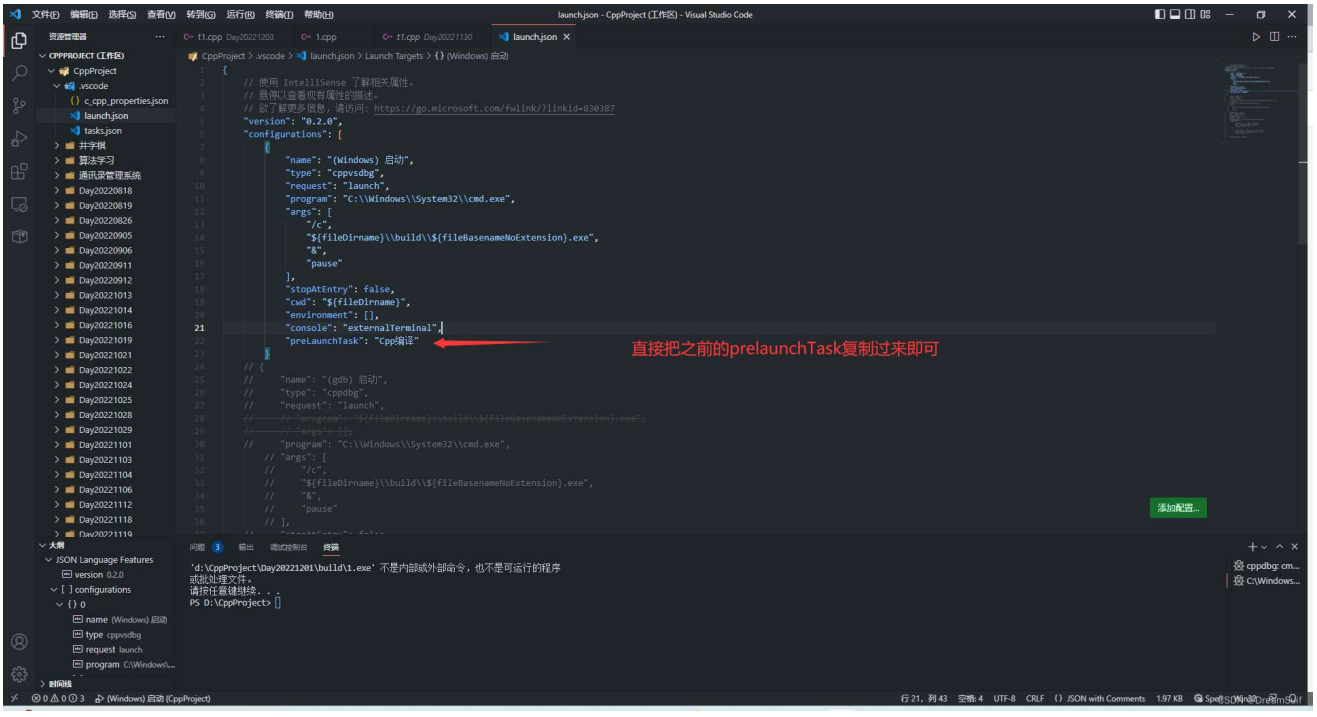
然后我们点击添加配置，选择windows启动，我们就会得到这样的一些内容：



program, args中的内容改成和原来的program，args中的内容一样，我们直接复制粘贴过来即可（这个内容不需要和我的一样，我只是给个示范）



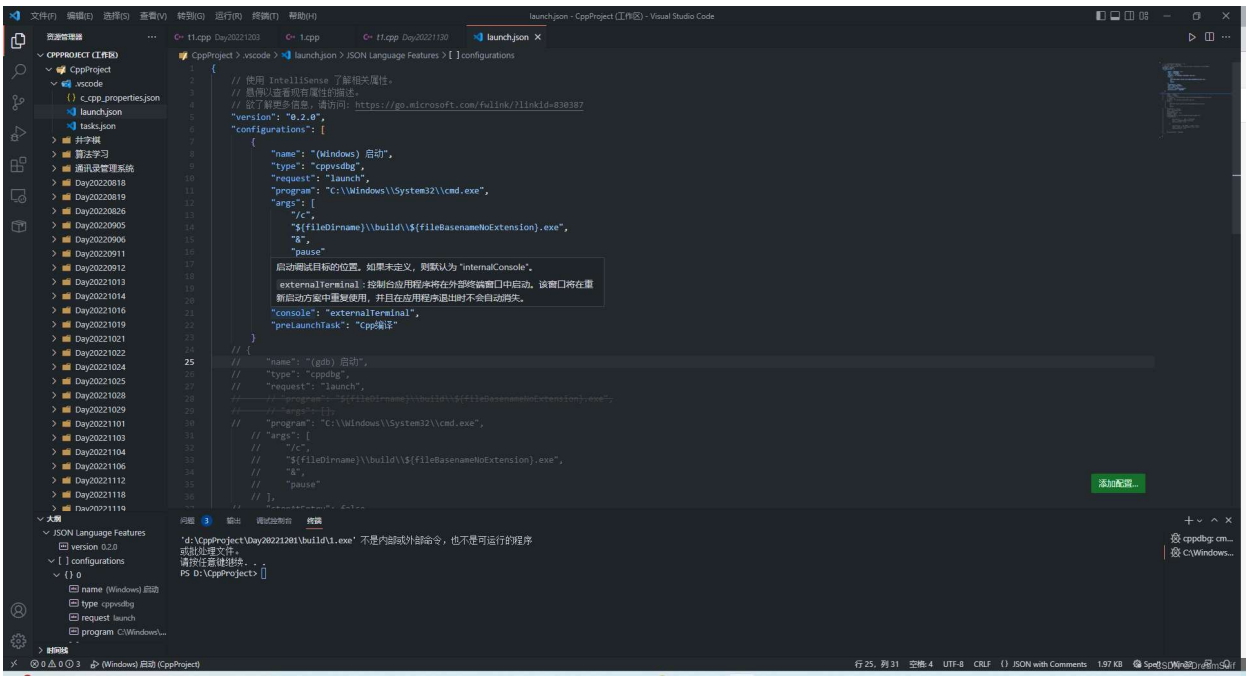
同样的我们需要调试前调用编译任务生成文件



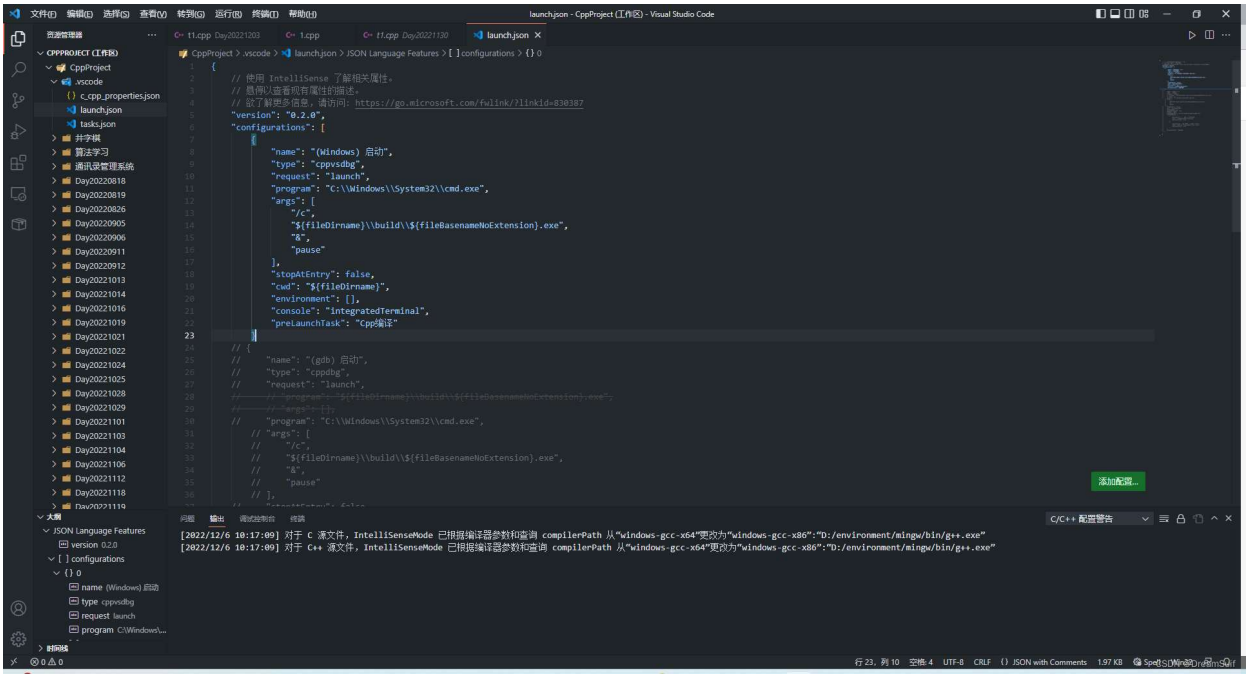
然后我们只需要改变console中的内容即可控制是在系统终端中运行程序还是在vscode终端中运行程序

该项值为 externalTerminal 则是在系统终端中运行程序

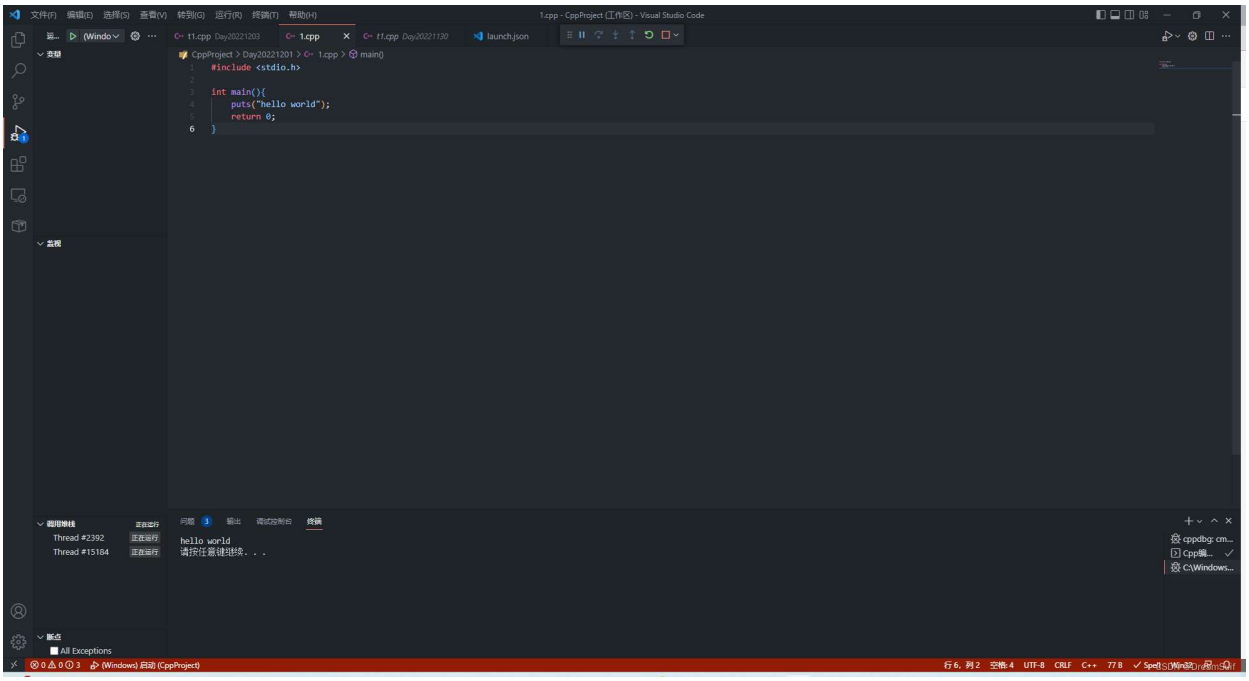
该项值为 integratedTerminal 则是在vscode终端中运行程序



我们把该项值改为 integratedTerminal 即可在vscode终端中运行程序



最终效果：


1.cpp - CppProject (工作区) - Visual Studio Code
t1.cpp Day20221203
1.cpp
t1.cpp Day20221130
launch.json
CppProject > Day20221201 > 1.cpp > main()
#include <stdio.h>
int main(){
puts("hello world");
return 0;
}
变量
监视
调用堆栈
Thread #2392
Thread #15184
断点
All Exceptions
问题 3
输出
调试控制台
终端
hello world
请按任意键继续. . .
cppdbg: cm...
Cpp编...
C:\Windows...
(Windows) 启动 (CppProject)
行 6，列 2
空格: 4
UTF-8
CRLF
C++
77 B