Debug Multi-files c++ program in Visual Studio Code

Lai Wei (SDS)

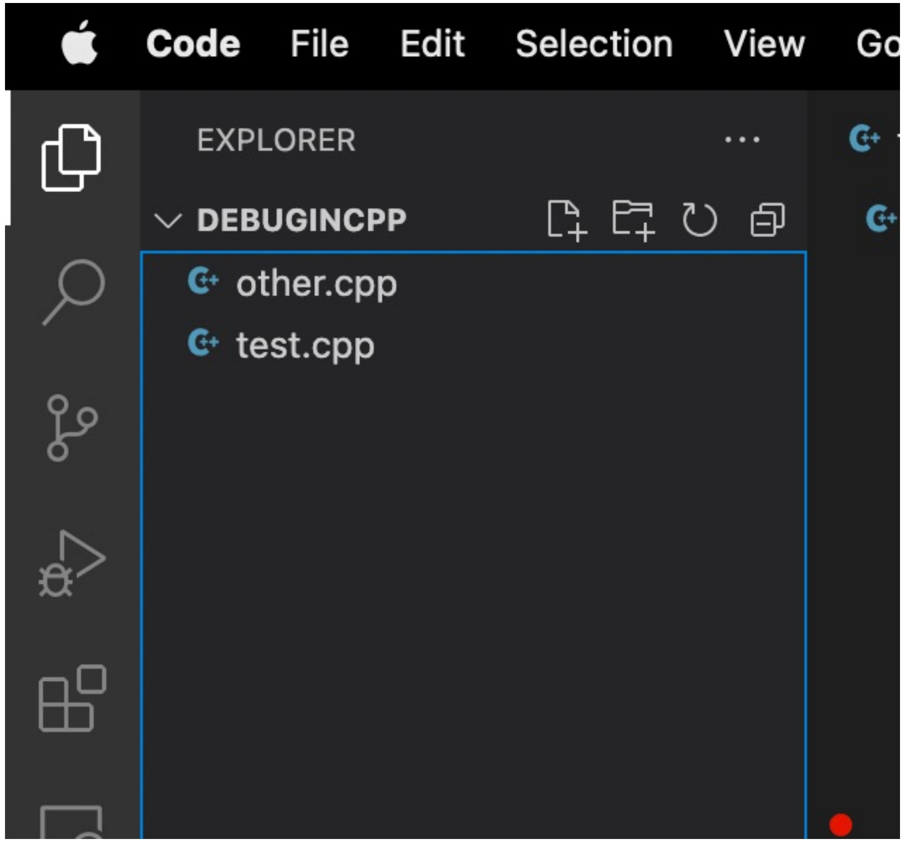
Nov. 12th, 2022

Introduction

- In CSC3002 so far, we are using VS Code with g++ compiler to run our cpp files. But how to debug the source file?
- This slides will show basic procedures to debug C++ program in Visual Studio Code
- This is totally an **optional topic**. It aims to let you know how to debug in Visual Studio Code, to make your programming experience better.

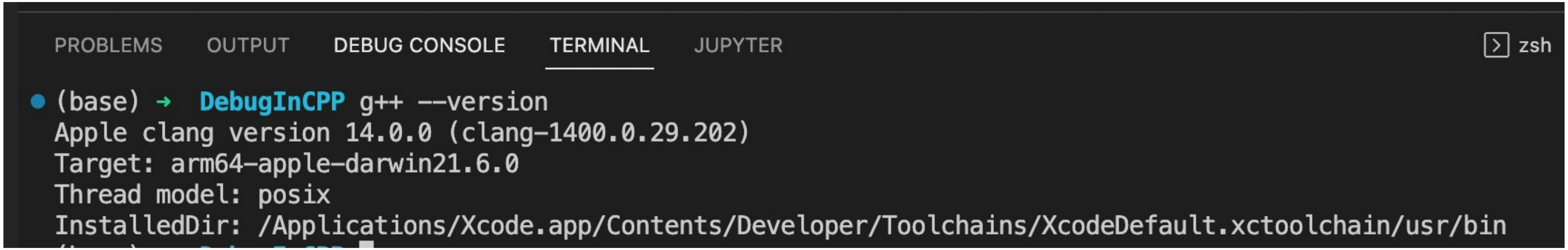
0. Pre-requisite

- You have just opened your program directory, not its parent directory.



0. Pre-requisite

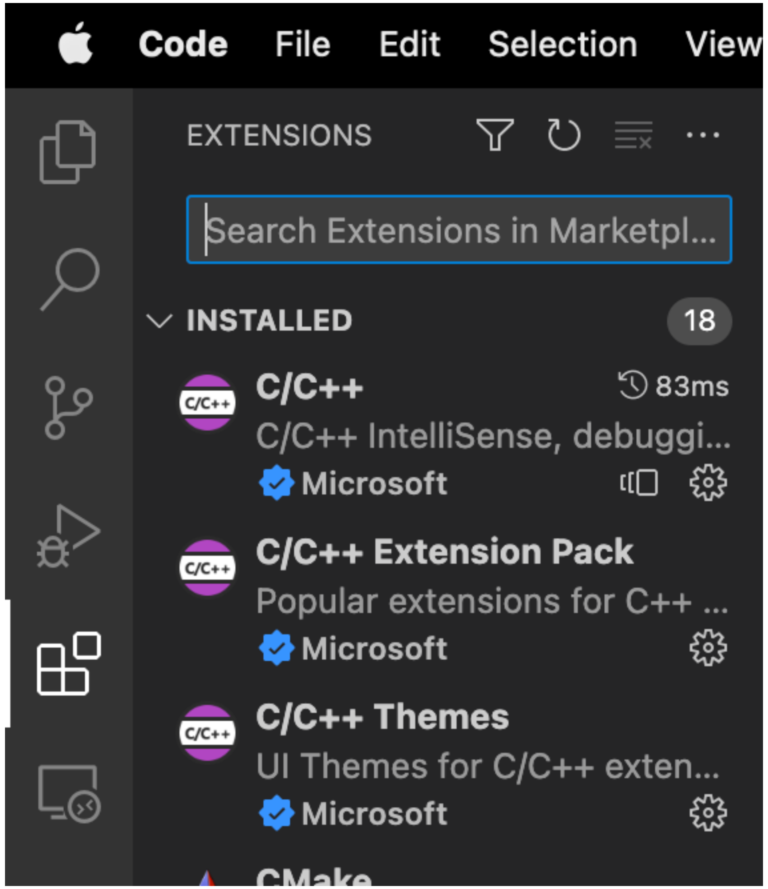
- You have g++ installed. ("g++ --version" gives some output like below)



```
● (base) →  DebugInCPP g++ --version
  Apple clang version 14.0.0 (clang-1400.0.29.202)
  Target: arm64-apple-darwin21.6.0
  Thread model: posix
  InstalledDir: /Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/bin
```

0. Pre-requisite

- You have VS Code-C++ extension installed.



C/C++ 83ms
C/C++ IntelliSense, debuggi...
Microsoft

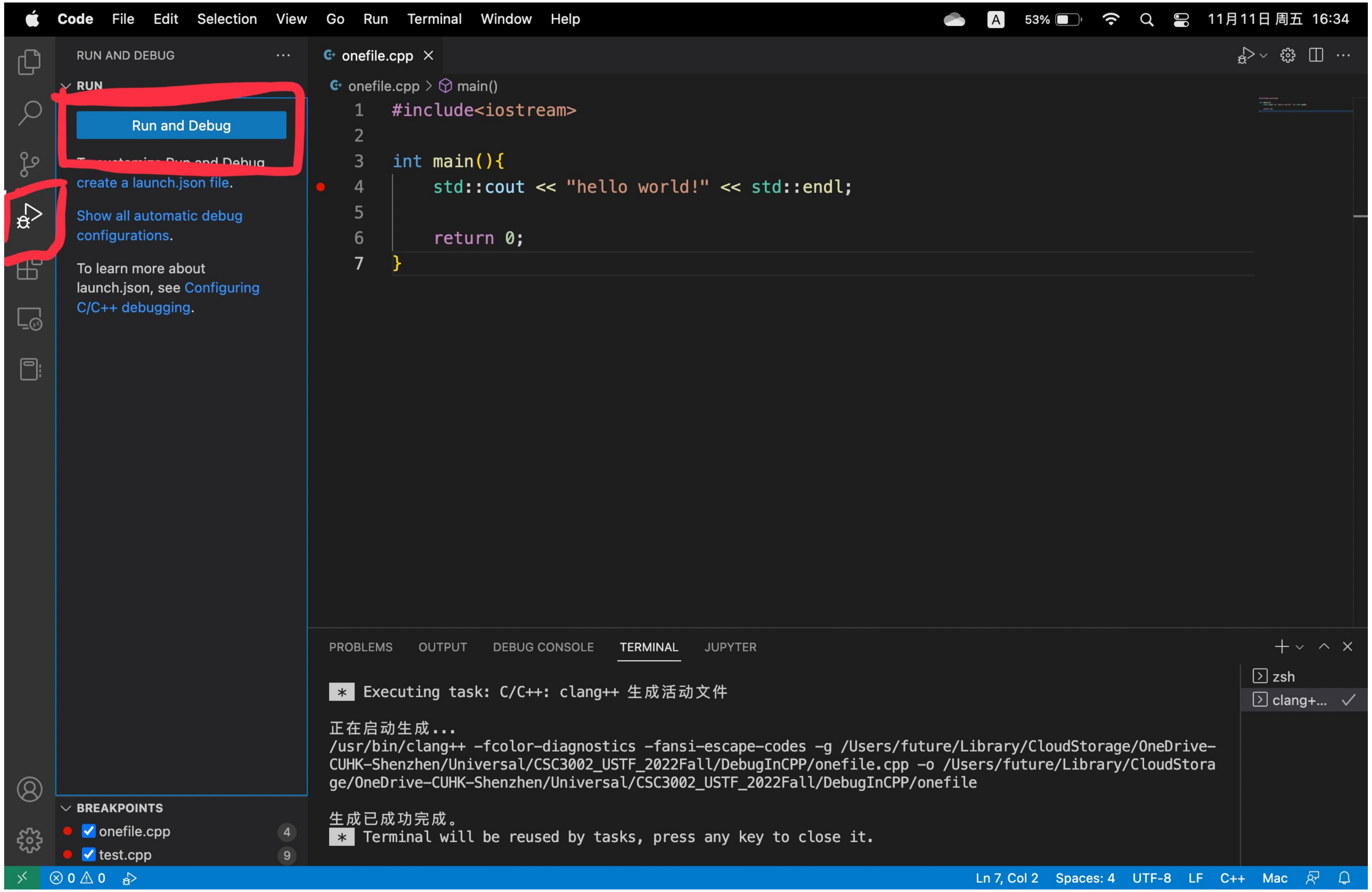
C/C++ Extension Pack
Popular extensions for C++ ...
Microsoft

C/C++ Themes
UI Themes for C/C++ exten...
Microsoft

CMake

1. Single-file program

- If your program only have one source .cpp file, you can debug by directing doing the following things:



To customize Run and Debug create a launch.json file.

Show all automatic debug configurations.

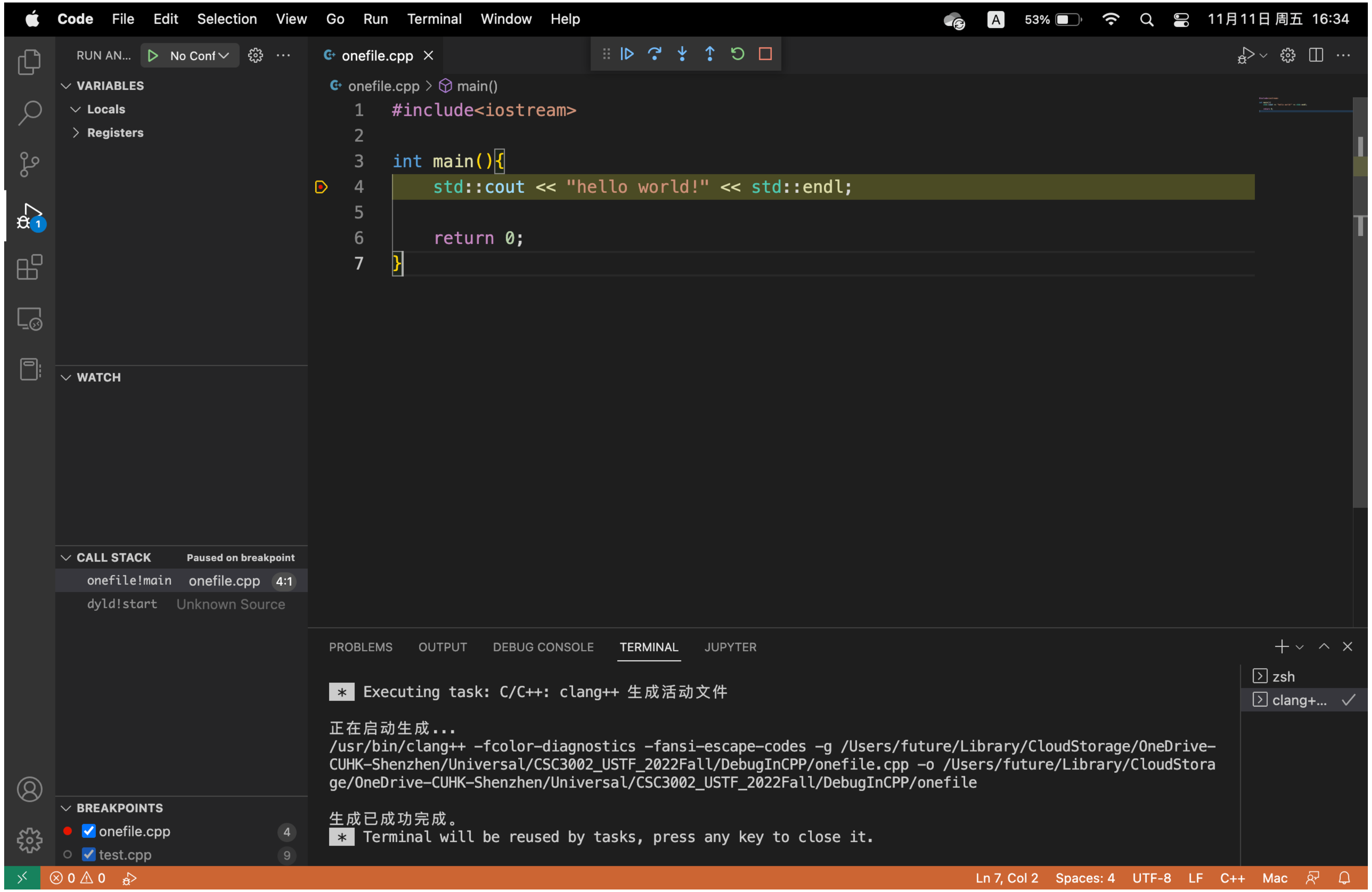
To learn more about launch.json, see Configuring C/C++ debugging.

```cpp
#include<iostream>

int main(){
    std::cout << "hello world!" << std::endl;

    return 0;
}
```

```
 * Executing task: C/C++: clang++ 生成活动文件

正在启动生成...
/usr/bin/clang++ -fcolor-diagnostics -fansi-escape-codes -g /Users/future/Library/CloudStorage/OneDrive-
CUHK-Shenzhen/Universal/CSC3002_USTF_2022Fall/DebugInCPP/onefile.cpp -o /Users/future/Library/CloudStora
ge/OneDrive-CUHK-Shenzhen/Universal/CSC3002_USTF_2022Fall/DebugInCPP/onefile

生成已成功完成。
 * Terminal will be reused by tasks, press any key to close it.
```

1. Single-file program

- The interface is just like what you do in writing Python.

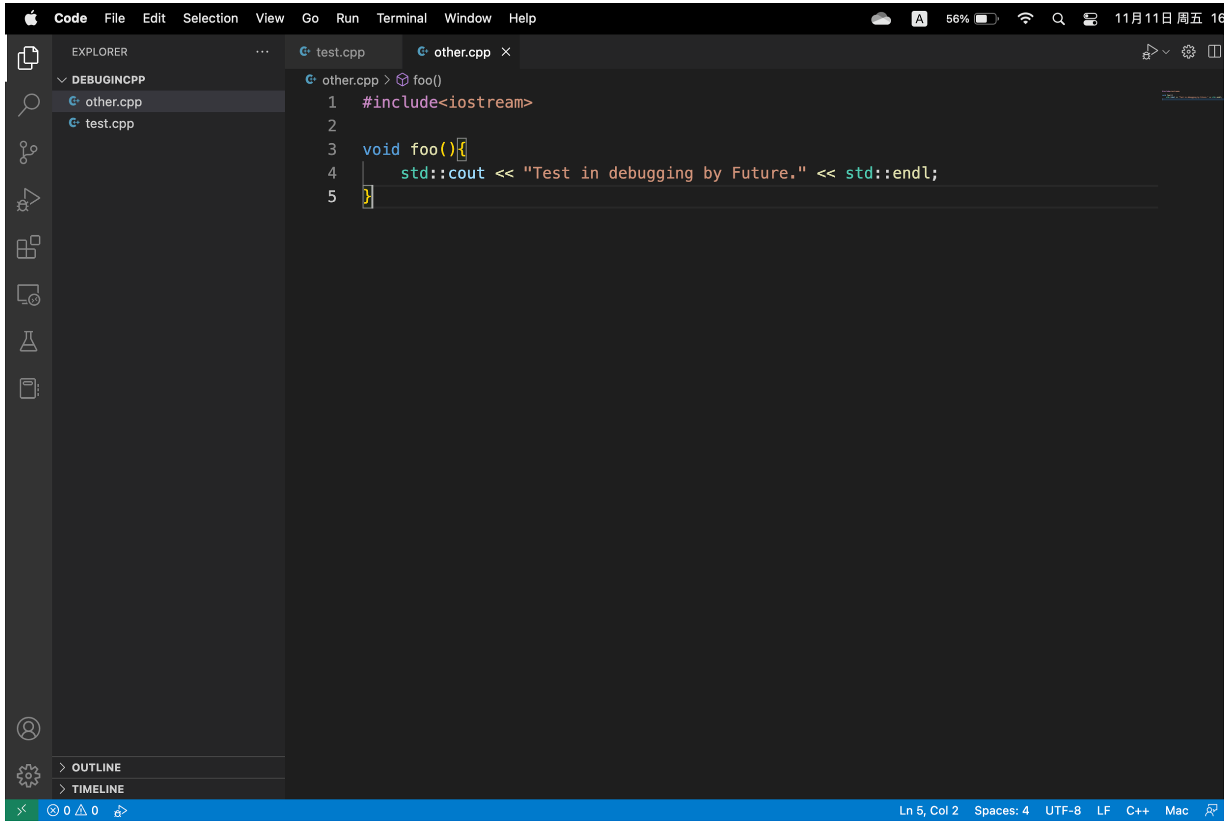


```cpp
#include<iostream>

int main(){
    std::cout << "hello world!" << std::endl;

    return 0;
}
```

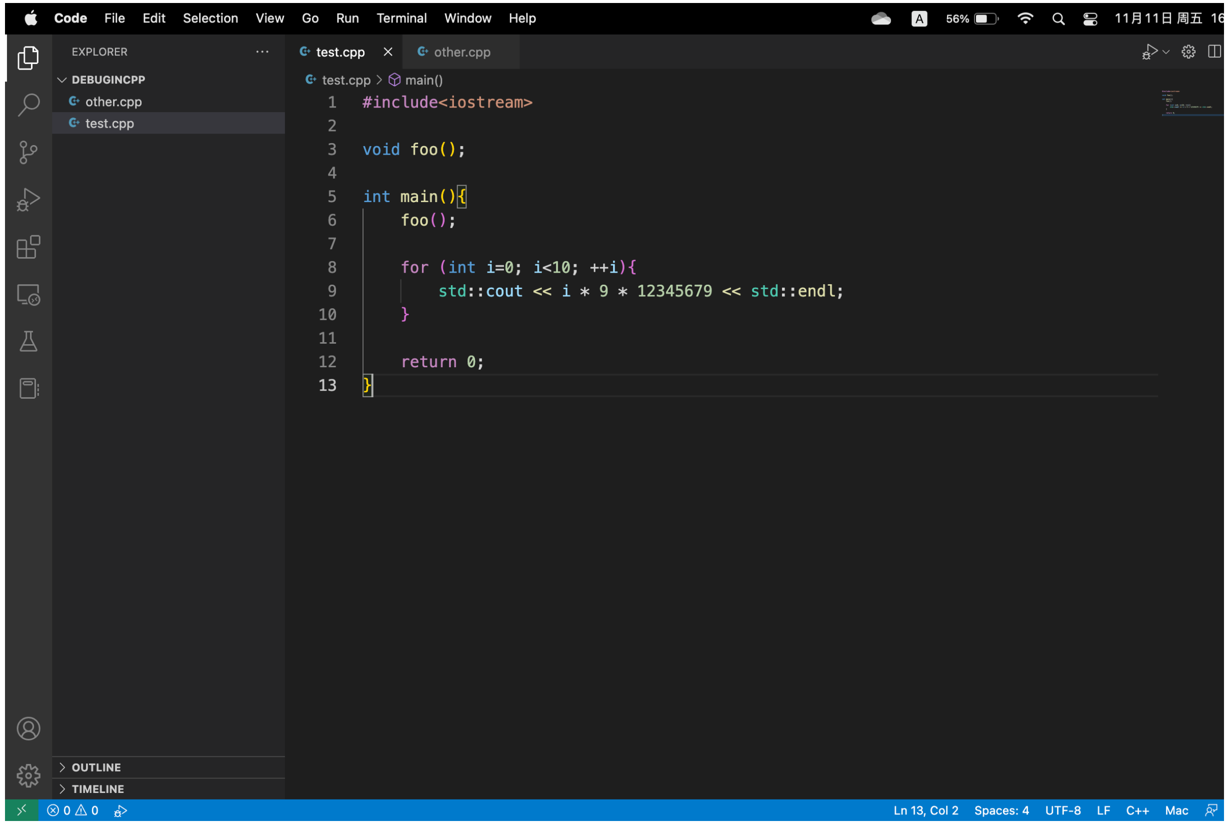
```
 * Executing task: C/C++: clang++ 生成活动文件

正在启动生成...
/usr/bin/clang++ -fcolor-diagnostics -fansi-escape-codes -g /Users/future/Library/CloudStorage/OneDrive-
CUHK-Shenzhen/Universal/CSC3002_USTF_2022Fall/DebugInCPP/onefile.cpp -o /Users/future/Library/CloudStora
ge/OneDrive-CUHK-Shenzhen/Universal/CSC3002_USTF_2022Fall/DebugInCPP/onefile

生成已成功完成。
 * Terminal will be reused by tasks, press any key to close it.
```

2. Multi-file program

Suppose we have such a two-file program, test.cpp and other.cpp.



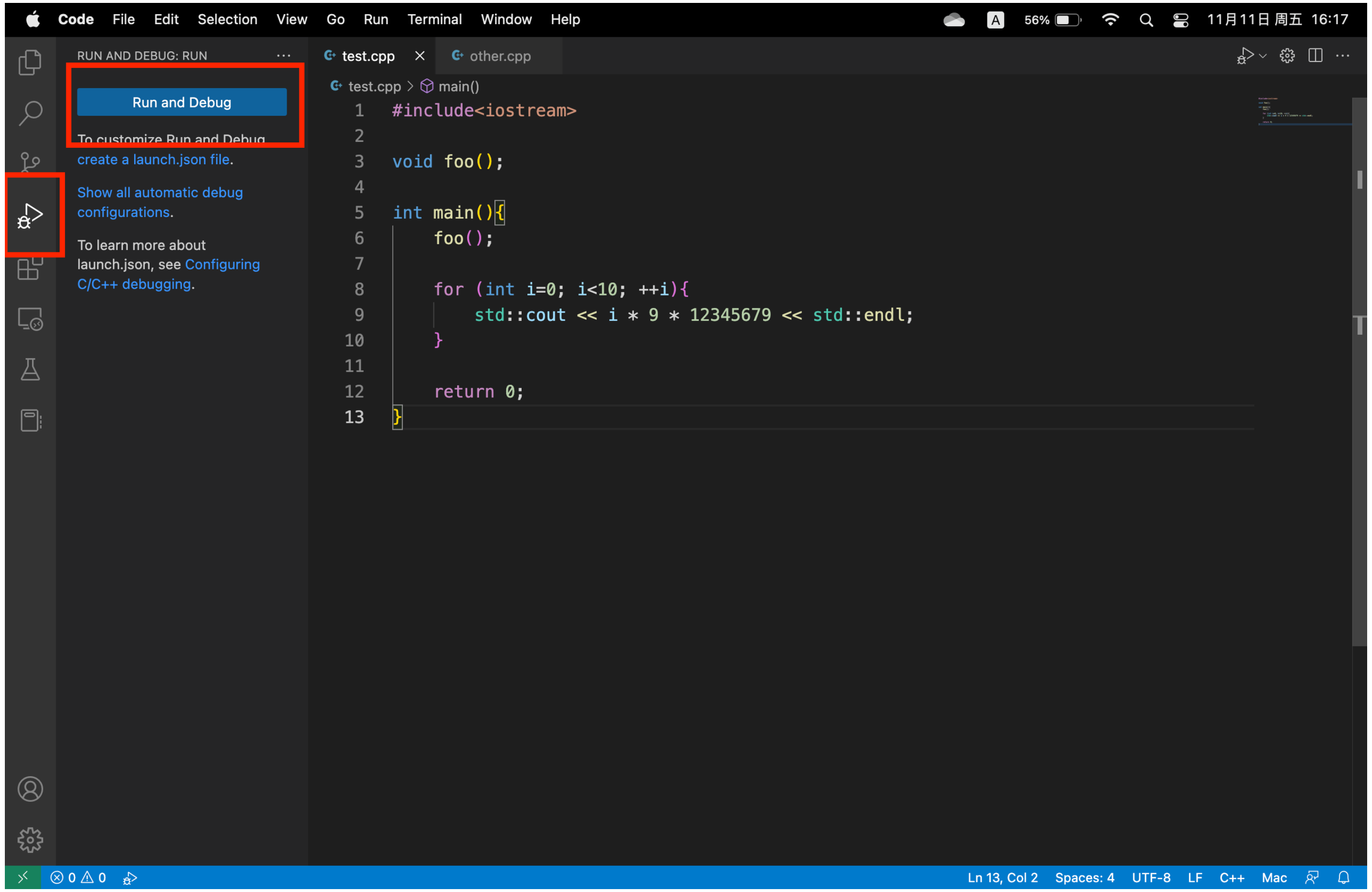
```cpp
#include<iostream>

void foo(){
    std::cout << "Test in debugging by Future." << std::endl;
}
```



```cpp
#include<iostream>

void foo();

int main(){
    foo();

    for (int i=0; i<10; ++i){
        std::cout << i * 9 * 12345679 << std::endl;
    }

    return 0;
}
```

2.1 Generating the debug config

- Go to the ”debug” side bar, clicking the “Run and Debug” once.



RUN AND DEBUG: RUN

Run and Debug

To customize Run and Debug create a launch.json file.

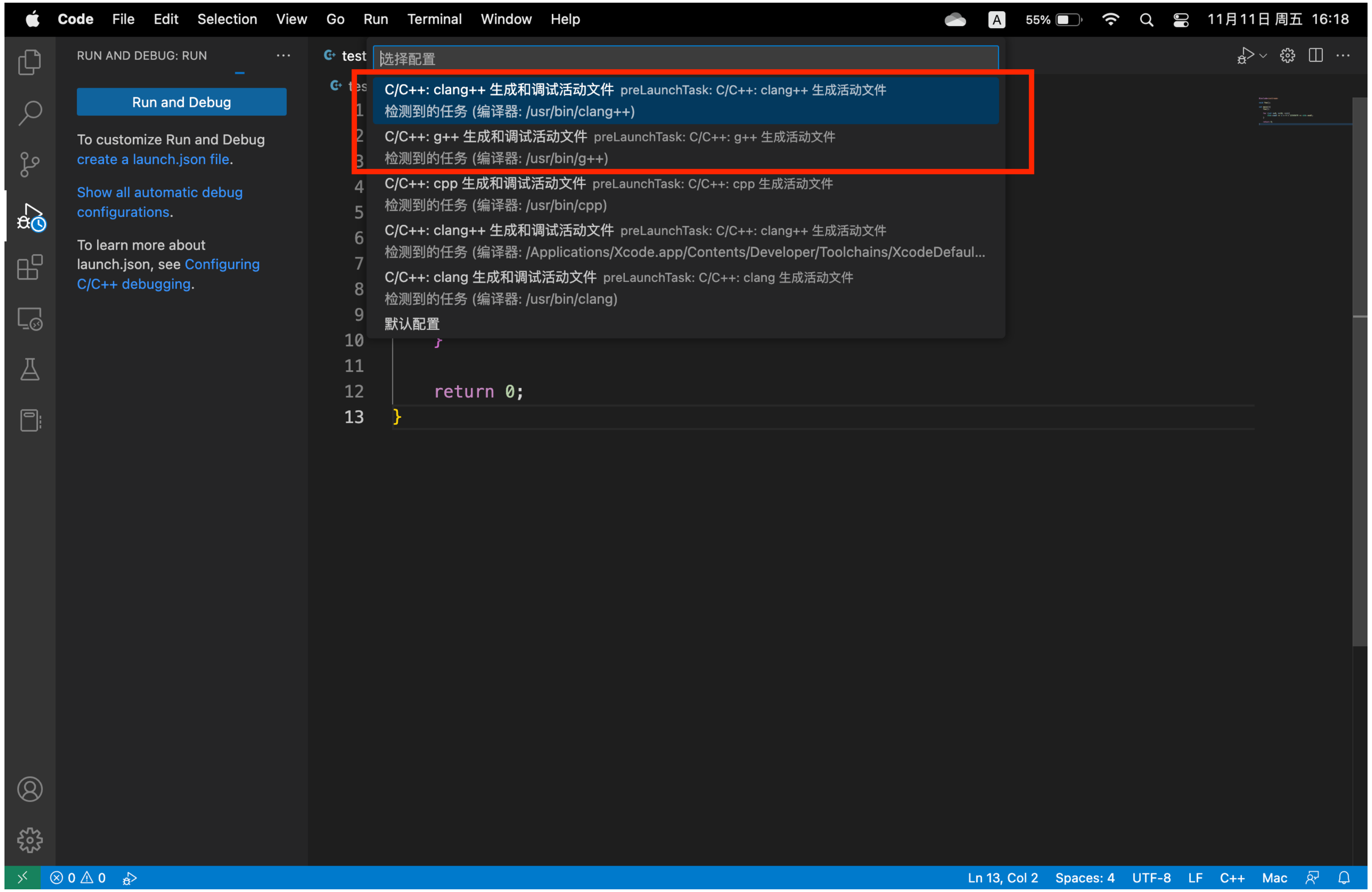
Show all automatic debug configurations.

To learn more about launch.json, see Configuring C/C++ debugging.

```
#include<iostream>

void foo();

int main(){
    foo();

    for (int i=0; i<10; ++i){
        std::cout << i * 9 * 12345679 << std::endl;
    }

    return 0;
}
```

2.1 Generating the debug config

- Select configuration “clang++”/”g++” (MacOS), or “g++” (Windows)



To customize Run and Debug create a launch.json file.

Show all automatic debug configurations.

To learn more about launch.json, see Configuring C/C++ debugging.

选择配置

C/C++: clang++ 生成和调试活动文件 preLaunchTask: C/C++: clang++ 生成活动文件
检测到的任务 (编译器: /usr/bin/clang++)

C/C++: g++ 生成和调试活动文件 preLaunchTask: C/C++: g++ 生成活动文件
检测到的任务 (编译器: /usr/bin/g++)

C/C++: cpp 生成和调试活动文件 preLaunchTask: C/C++: cpp 生成活动文件
检测到的任务 (编译器: /usr/bin/cpp)

C/C++: clang++ 生成和调试活动文件 preLaunchTask: C/C++: clang++ 生成活动文件
检测到的任务 (编译器: /Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefaul...

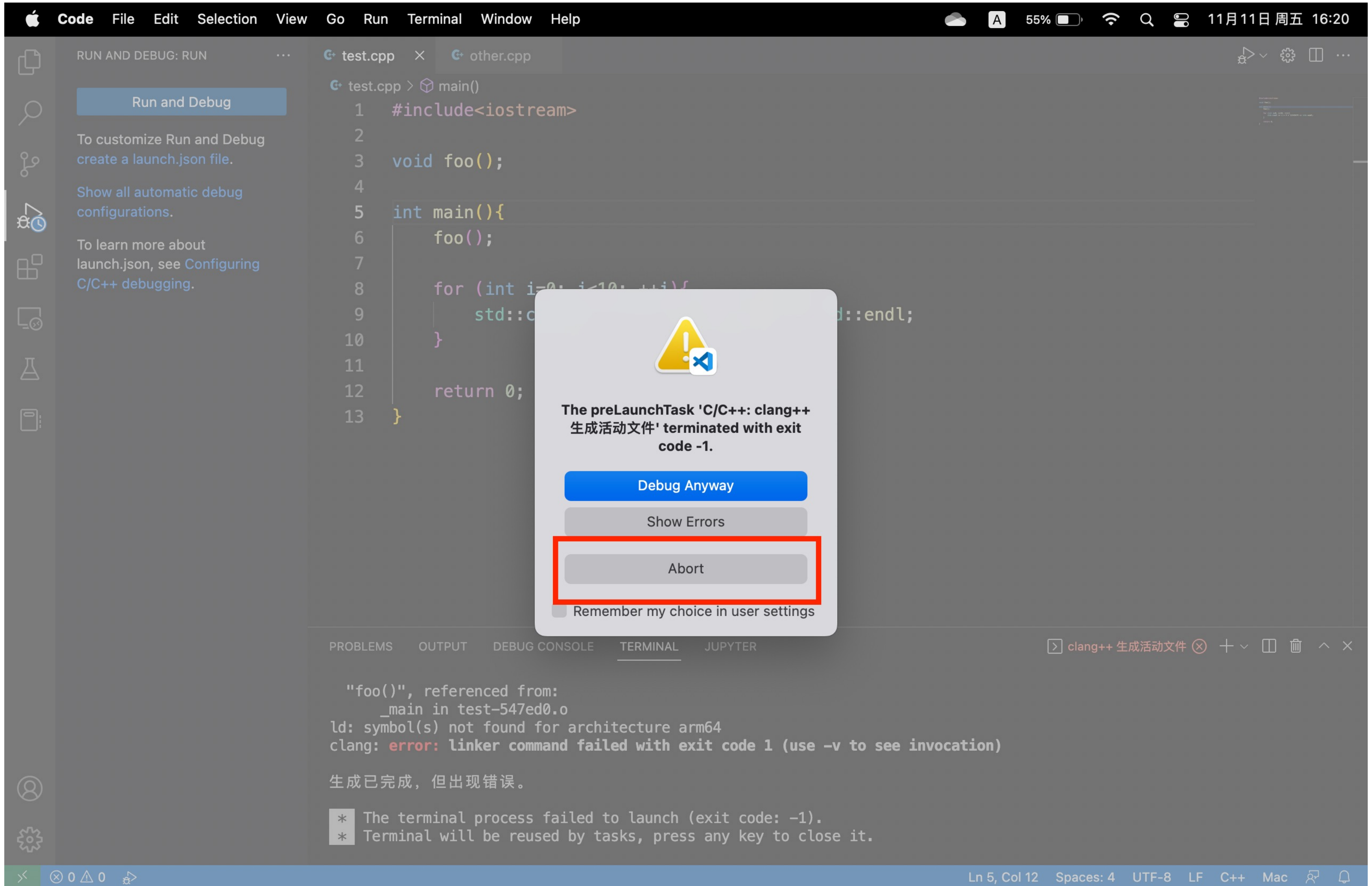
C/C++: clang 生成和调试活动文件 preLaunchTask: C/C++: clang 生成活动文件
检测到的任务 (编译器: /usr/bin/clang)

默认配置

```
    }

    return 0;
}
```

2.1 Generating the debug config

- You will see it returns an error. Click “abort” first.



Code File Edit Selection View Go Run Terminal Window Help
55%
11月11日 周五 16:20
RUN AND DEBUG: RUN
Run and Debug
test.cpp
other.cpp
test.cpp > main()
Debug Anyway
Show Errors
Abort
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
clang++ 生成活动文件
Ln 5, Col 12 Spaces: 4 UTF-8 LF C++ Mac

To customize Run and Debug create a launch.json file.

Show all automatic debug configurations.

To learn more about launch.json, see Configuring C/C++ debugging.

```
#include<iostream>

void foo();

int main(){
    foo();

    for (int i=0; i<10; ++i){
        std::c                                  d::endl;
    }

    return 0;
}
```

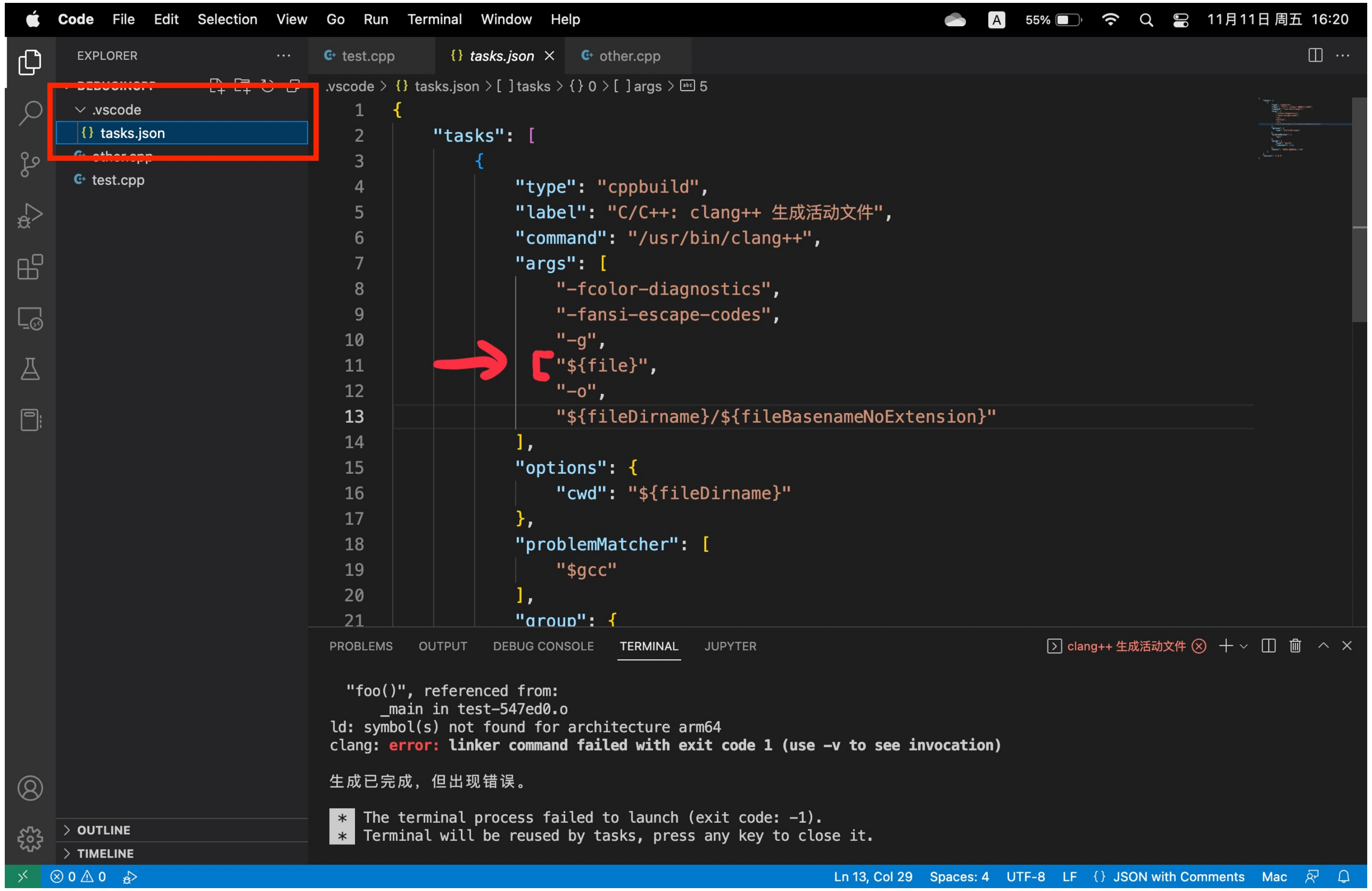
The preLaunchTask 'C/C++: clang++ 生成活动文件' terminated with exit code -1.

□ Remember my choice in user settings

```
  "foo()", referenced from:
      _main in test-547ed0.o
ld: symbol(s) not found for architecture arm64
clang: error: linker command failed with exit code 1 (use -v to see invocation)

生成已完成，但出现错误。

 *  The terminal process failed to launch (exit code: -1).
 *  Terminal will be reused by tasks, press any key to close it.
```

2.1 Generating the debug config

- But, you will find that we have an “tasks.json” configuration generated in the folder “.vscode”. Edit this file:

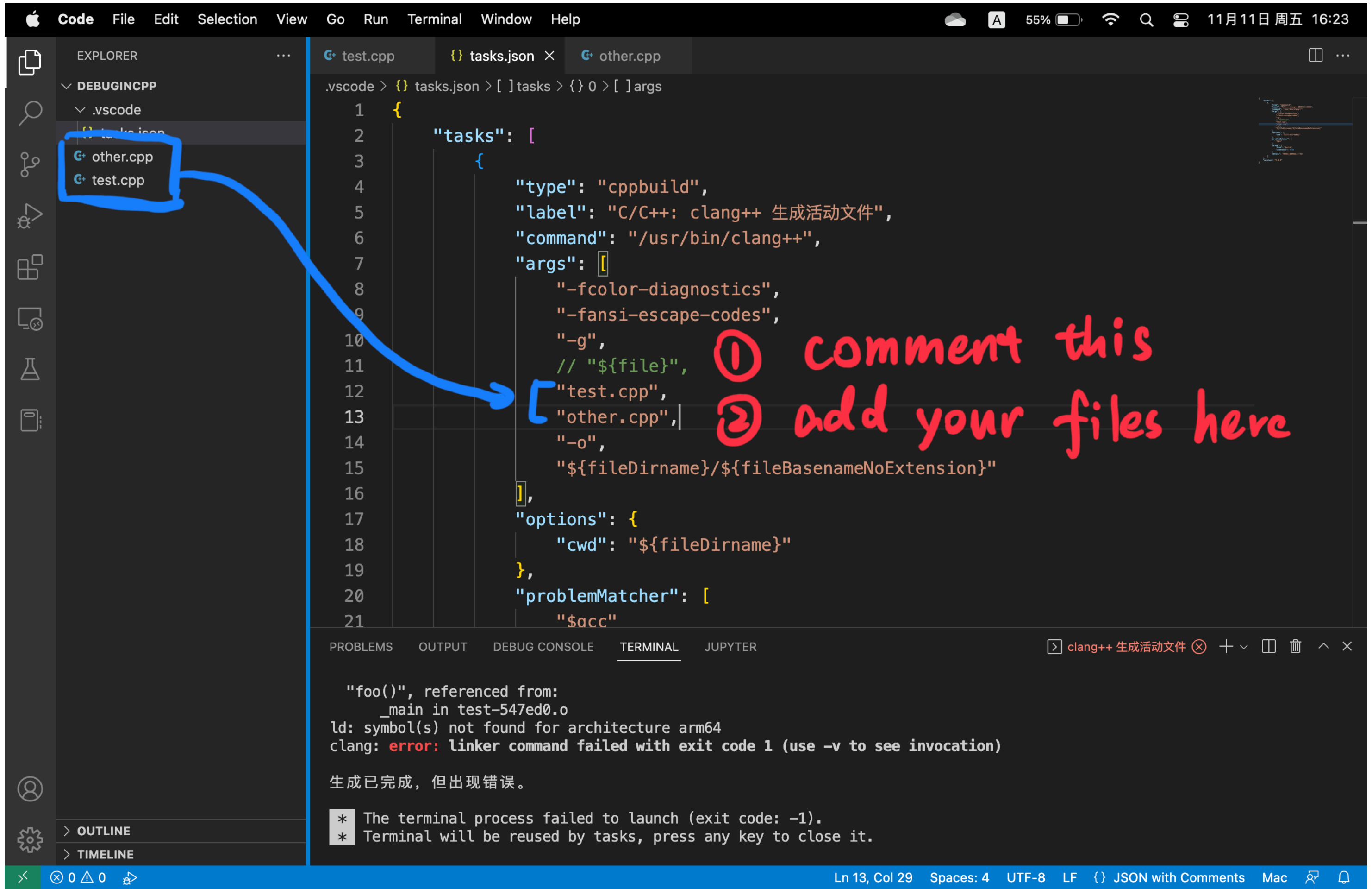


```
{
    "tasks": [
        {
            "type": "cppbuild",
            "label": "C/C++: clang++ 生成活动文件",
            "command": "/usr/bin/clang++",
            "args": [
                "-fcolor-diagnostics",
                "-fansi-escape-codes",
                "-g",
                "${file}",
                "-o",
                "${fileDirname}/${fileBasenameNoExtension}"
            ],
            "options": {
                "cwd": "${fileDirname}"
            },
            "problemMatcher": [
                "$gcc"
            ],
            "group": {
```

```
  "foo()", referenced from:
      _main in test-547ed0.o
ld: symbol(s) not found for architecture arm64
clang: error: linker command failed with exit code 1 (use -v to see invocation)

生成已完成，但出现错误。

 *  The terminal process failed to launch (exit code: -1).
 *  Terminal will be reused by tasks, press any key to close it.
```

2.2 Editing the debug config

- Comment (delete) the “${file}”, and add your program source files one by one. E.g., “test.cpp”, “other.cpp” in my case. Save the .json file.



```
{
    "tasks": [
        {
            "type": "cppbuild",
            "label": "C/C++: clang++ 生成活动文件",
            "command": "/usr/bin/clang++",
            "args": [
                "-fcolor-diagnostics",
                "-fansi-escape-codes",
                "-g",
                // "${file}",
                "test.cpp",
                "other.cpp",
                "-o",
                "${fileDirname}/${fileBasenameNoExtension}"
            ],
            "options": {
                "cwd": "${fileDirname}"
            },
            "problemMatcher": [
                "$gcc"
```

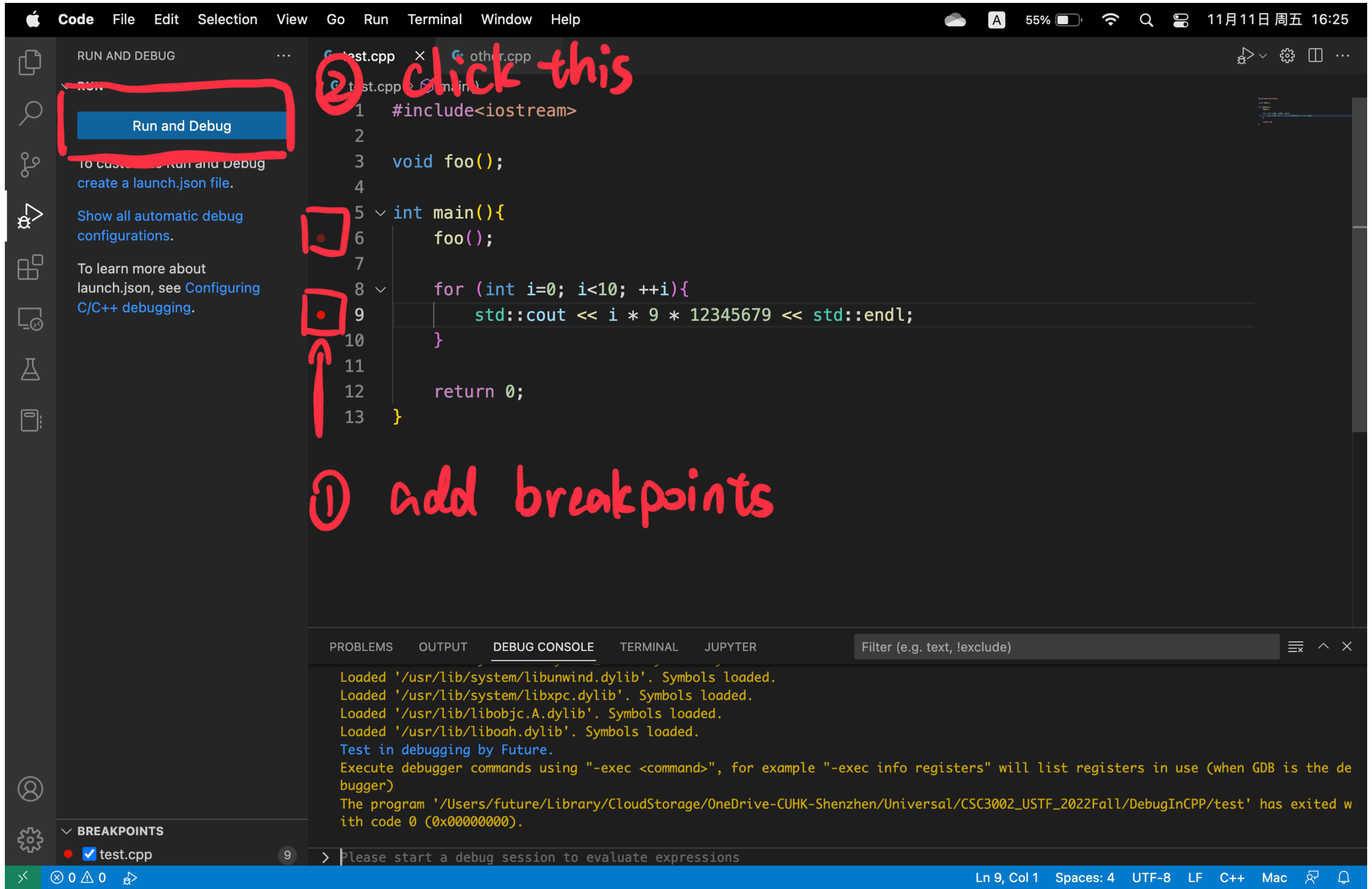
① comment this

② add your files here

```
  "foo()", referenced from:
      _main in test-547ed0.o
ld: symbol(s) not found for architecture arm64
clang: error: linker command failed with exit code 1 (use -v to see invocation)

生成已完成，但出现错误。

 *  The terminal process failed to launch (exit code: -1).
 *  Terminal will be reused by tasks, press any key to close it.
```

2.3 Debug your program

- If everything goes smooth, return to your code, add some breakpoints, and click “Run and Debug” on the left side bar.



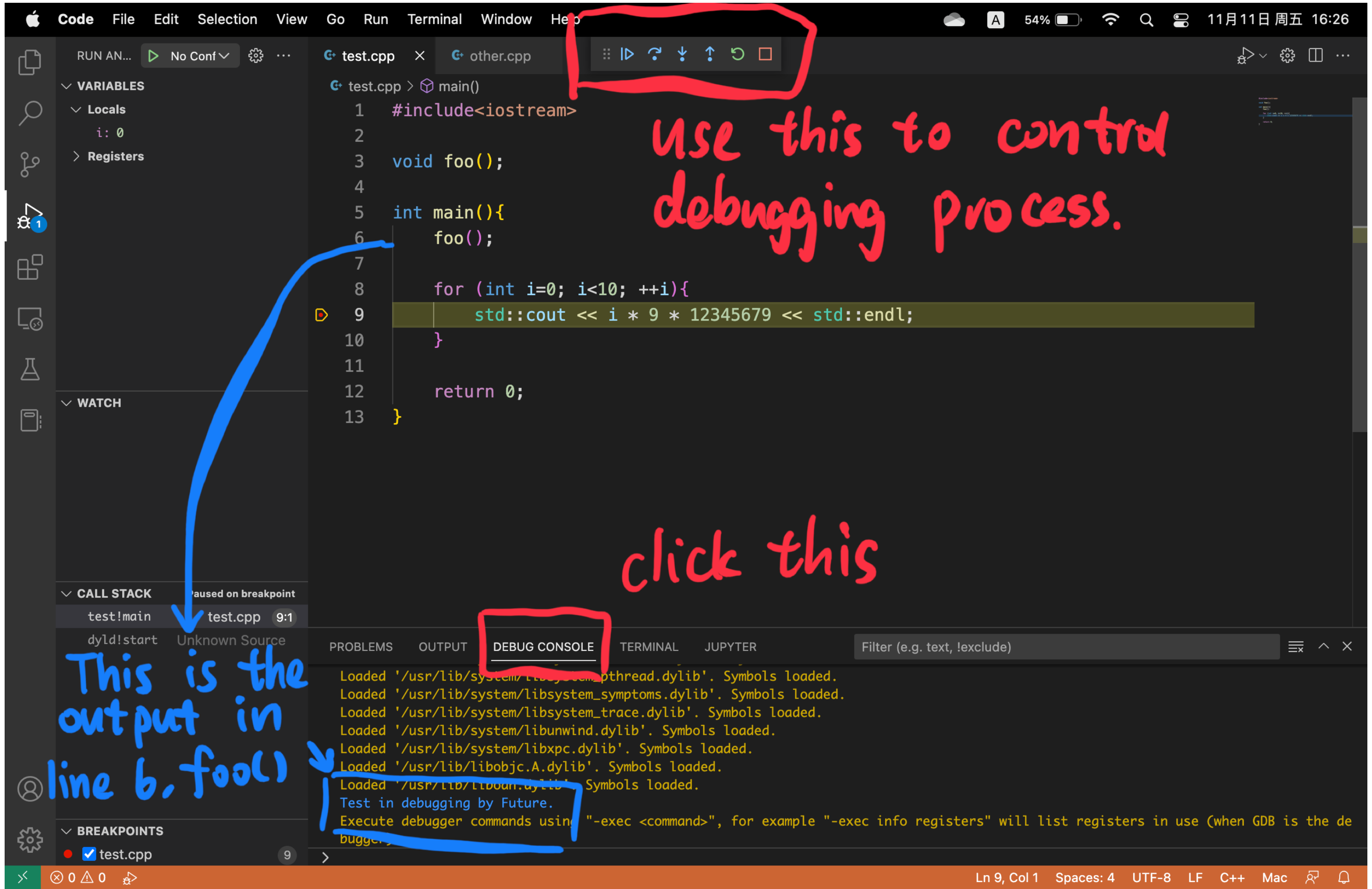
Code File Edit Selection View Go Run Terminal Window Help
55%
11月11日 周五 16:25
RUN AND DEBUG
RUN
Run and Debug
To customize Run and Debug create a launch.json file.
Show all automatic debug configurations.
To learn more about launch.json, see Configuring C/C++ debugging.
BREAKPOINTS
test.cpp
test.cpp
other.cpp
test.cpp > main()
② click this
① add breakpoints
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
JUPYTER
Filter (e.g. text, !exclude)
Ln 9, Col 1 Spaces: 4 UTF-8 LF C++ Mac

```cpp
#include<iostream>

void foo();

int main(){
    foo();

    for (int i=0; i<10; ++i){
        std::cout << i * 9 * 12345679 << std::endl;
    }

    return 0;
}
```

```
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/liboah.dylib'. Symbols loaded.
Test in debugging by Future.
Execute debugger commands using "-exec <command>", for example "-exec info registers" will list registers in use (when GDB is the de
bugger)
The program '/Users/future/Library/CloudStorage/OneDrive-CUHK-Shenzhen/Universal/CSC3002_USTF_2022Fall/DebugInCPP/test' has exited w
ith code 0 (0x00000000).
> Please start a debug session to evaluate expressions
```

2.3 Debug your program

- If everything goes smooth, you will enter the debugging interface. The output is at the "DEBUG CONSOLE" below.



```
1   #include<iostream>
2
3   void foo();
4
5   int main(){
6       foo();
7
8       for (int i=0; i<10; ++i){
9           std::cout << i * 9 * 12345679 << std::endl;
10      }
11
12      return 0;
13  }
```

```
Loaded '/usr/lib/system/libsystem_pthread.dylib'. Symbols loaded.
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/liboah.dylib'. Symbols loaded.
Test in debugging by Future.
Execute debugger commands using "-exec <command>", for example "-exec info registers" will list registers in use (when GDB is the de
bugger)
```

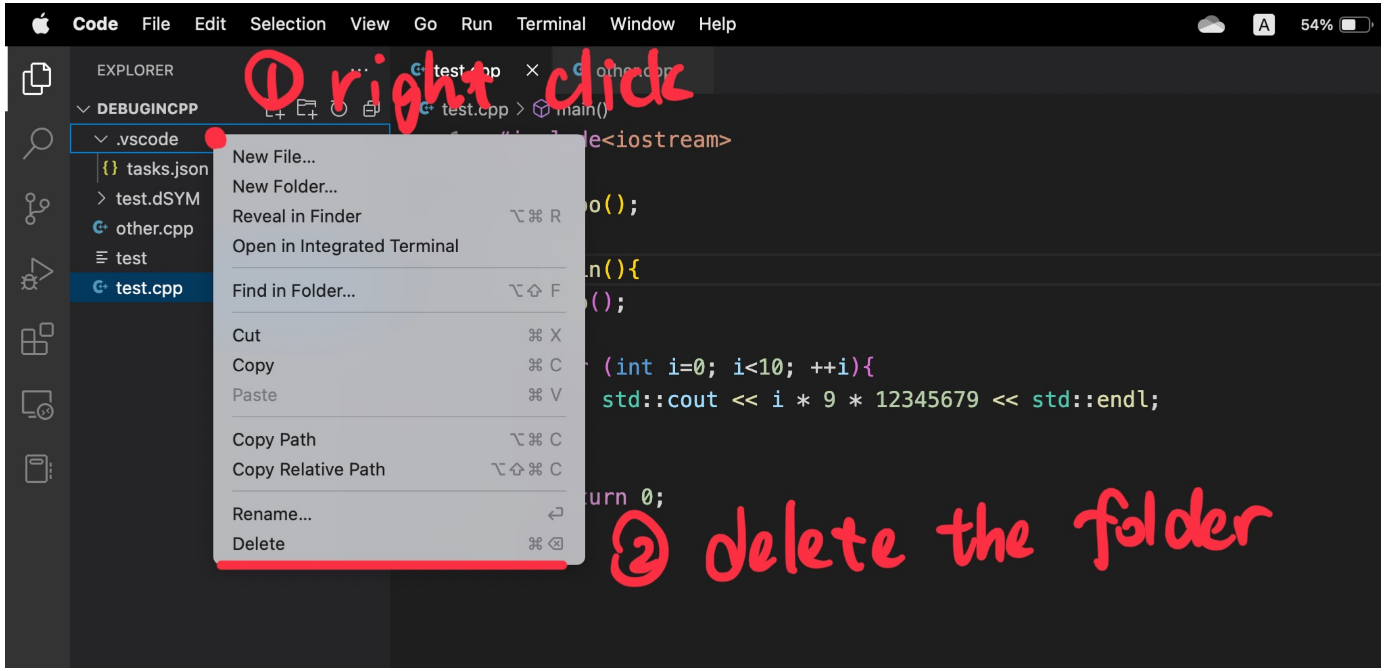
use this to control debugging process.

click this

This is the output in line 6, foo()

3 Important Notes

- If you want to debug **another program**, you need to **change the ".vscode/tasks.json" file manually**. Change the source file names to your new program.
- If anything goes wrong, **delete the ".vscode" file and try it again**. If still fails, you may search online for solutions, or use "std::cout" to debug, haha.



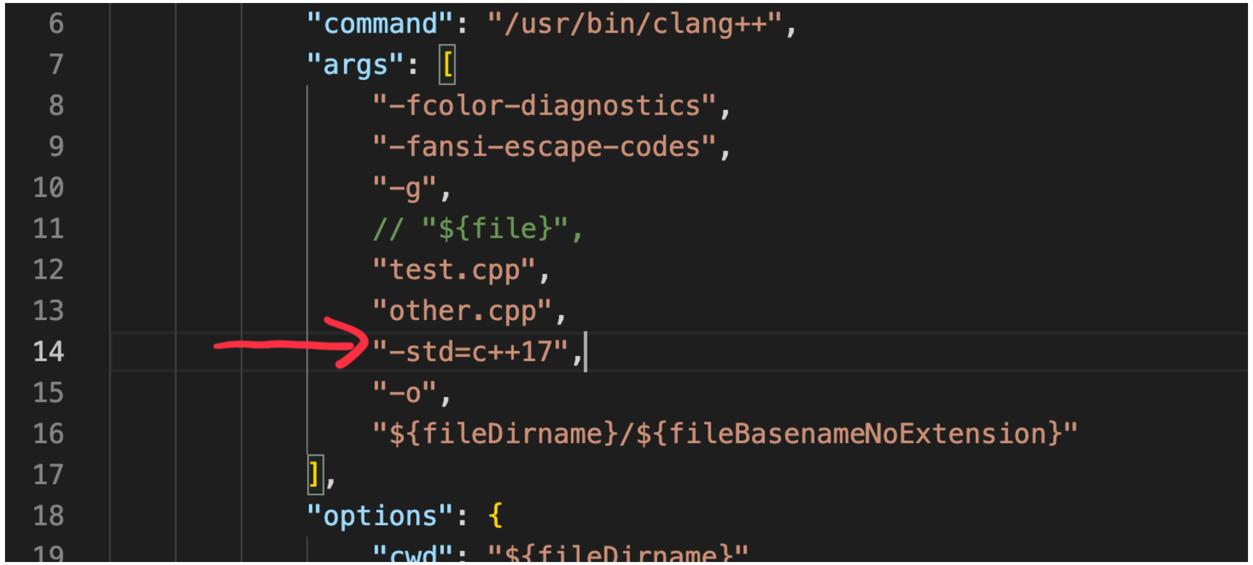
```
le<iostream>

o();

n(){
();

 (int i=0; i<10; ++i){
  std::cout << i * 9 * 12345679 << std::endl;

urn 0;
```

① right click

② delete the folder

3 Important Notes

- To indicate that your desired C++ standard is c++ 17, you may also add one item in “tasks.json”:



```
            "command": "/usr/bin/clang++",
            "args": [
                "-fcolor-diagnostics",
                "-fansi-escape-codes",
                "-g",
                // "${file}",
                "test.cpp",
                "other.cpp",
                "-std=c++17",
                "-o",
                "${fileDirname}/${fileBasenameNoExtension}"
            ],
            "options": {
                "cwd": "${fileDirname}"
```

- A demo video is also available on BB.
- No response to personal emails asking ”why I can’t debug multi-files in my VS Code”. Please search it online first. This is an optional topic.