Python & C++ Program Design

Visual Studio Code

-- vscode: hello world!

Junjie Cao @ DLUT

Summer 2022

https://github.com/jjcao-school/c

Why vscode?

As usual, everyone was using the CodeBlocks IDE and Visual Studio IDE. But I was already used to Visual Studio Code

- A lightweight editor
- Versatile: c++, python, html, markdown, latex, …
- Powerful
- Cross-platform

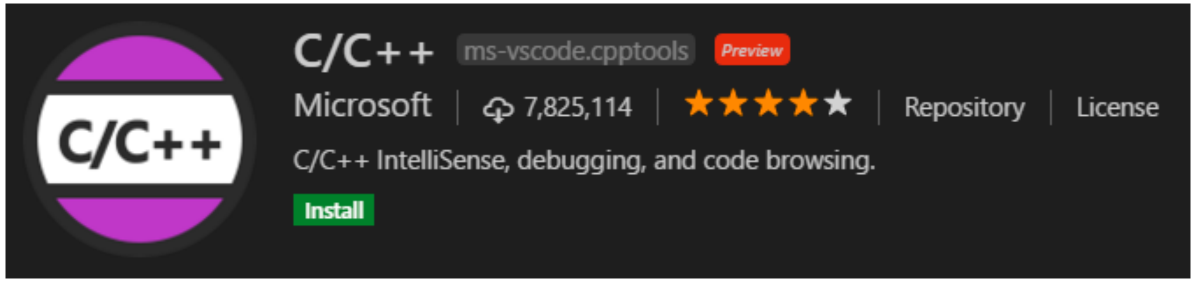
Prerequisites

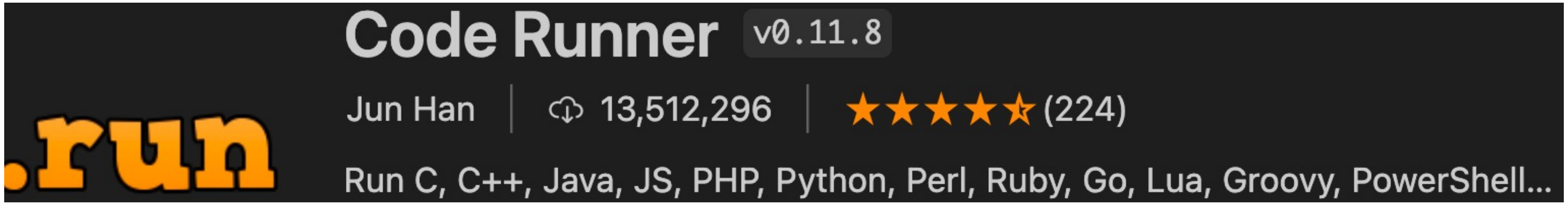
- VS Code
- VS Code Python extension
- Python 3

- https://code.visualstudio.com/docs/python/python-tutorial

大纲

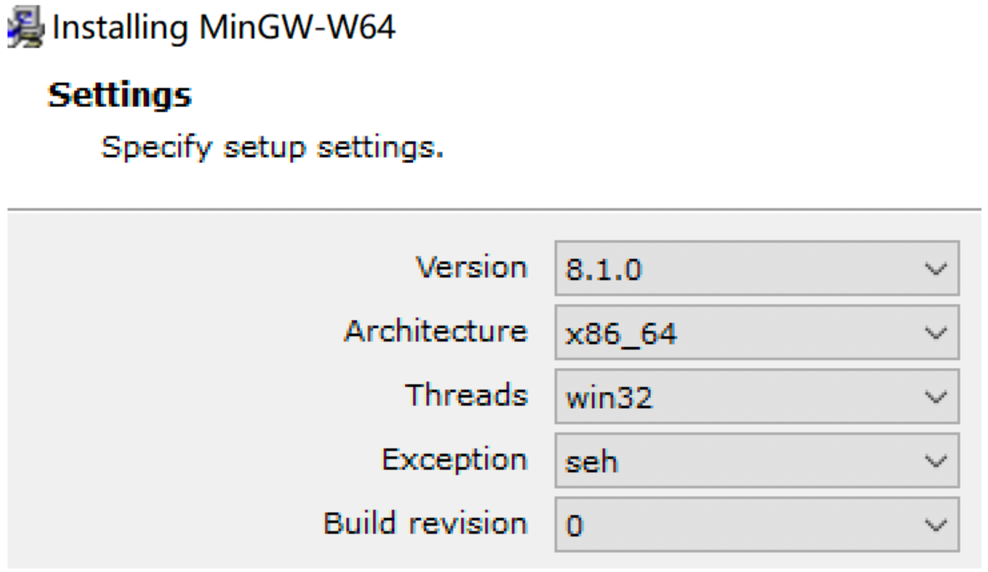
1. Install IDE: vscode
2. Install c++ compiler
    1. MinGW for windows
    2. Clang for mac
3. Install the "**ms-vscode.cpptools**" extension
4. Install “Code Runner” extension
5. Create hello.cpp and edit it
6. Run it
7. Check the project folder
8. Debug it


C/C++ ms-vscode.cpptools Preview
Microsoft | 7,825,114 | ★★★★★ | Repository | License
C/C++ IntelliSense, debugging, and code browsing.
Install


Code Runner v0.11.8
Jun Han | 13,512,296 | ★★★★★ (224)
Run C, C++, Java, JS, PHP, Python, Perl, Ruby, Go, Lua, Groovy, PowerShell...

Install MinGW for Windows

- https://code.visualstudio.com/docs/cpp/config-mingw



Installing MinGW-W64

Settings

Specify setup settings.

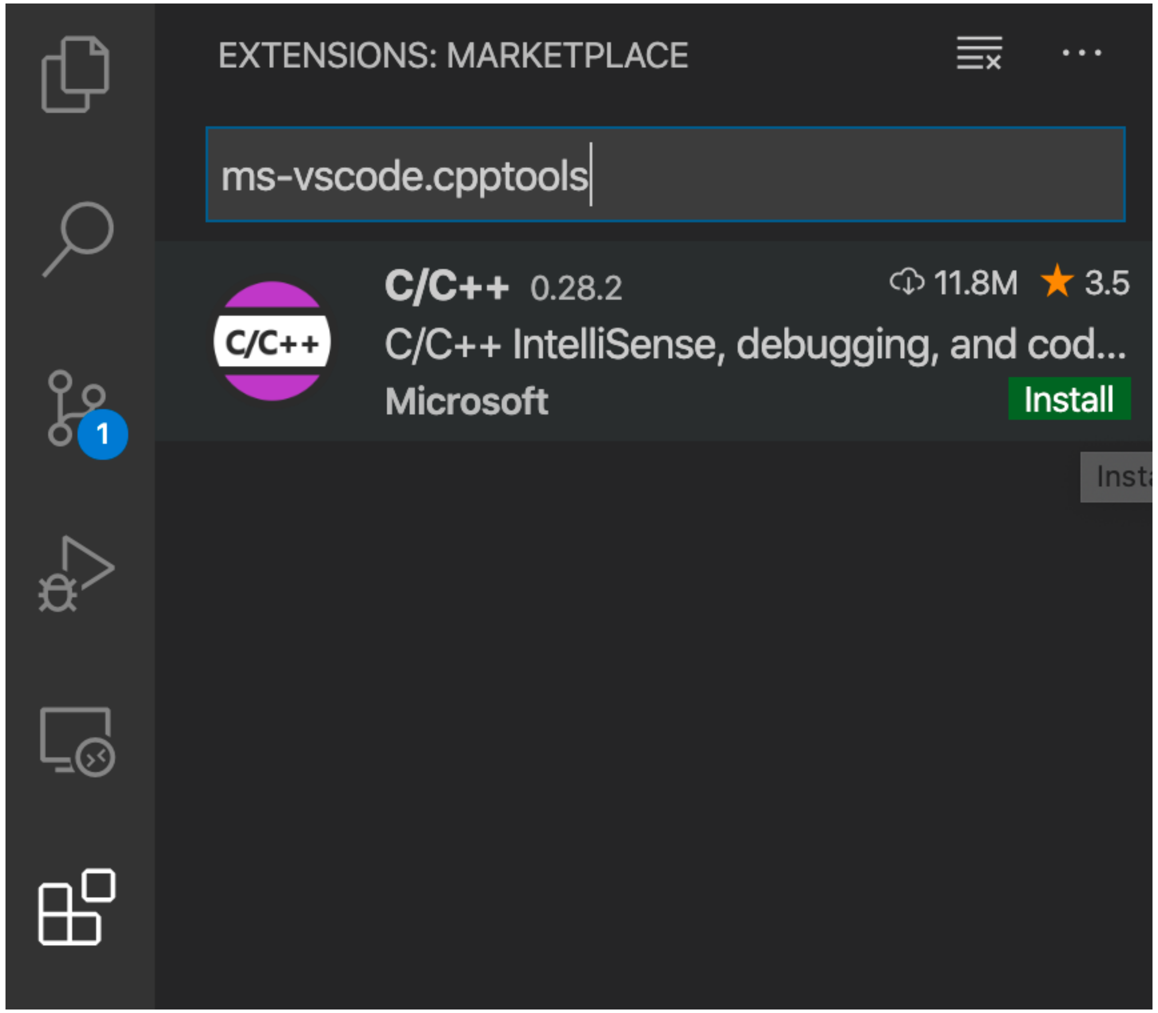
Version	8.1.0
Architecture	x86_64
Threads	win32
Exception	seh
Build revision	0

- Maybe you can stop after “Check your MinGW installation”

Install clang for Mac

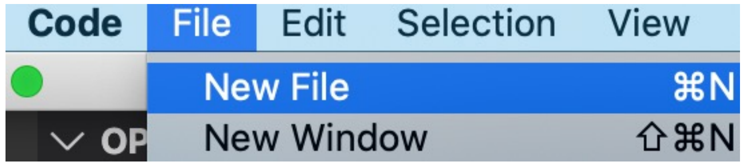
- xcode-select --install

Install the "ms-vscode.cpptools" extension



To create the first program

- select ”**File**" > "**New file**". This will open a new file window.



- Save the file ("**File**" > "**Save**") into a new directory.
    - You can name the directory anything you want, but this example will call the directory ”**c_labs**" and the file ”**hello.cpp**".

Write the actual program

```
#include <iostream>
int main() {
// Output the hello world text
std::cout << "Hello world!" << std::endl;
return 0;
}
```

Run code

① Click the triangle at upper right corner

② This opens a OUTPUT window in the lower portion of the IDE.

③ Inside this window, we found

  ① [Running] …

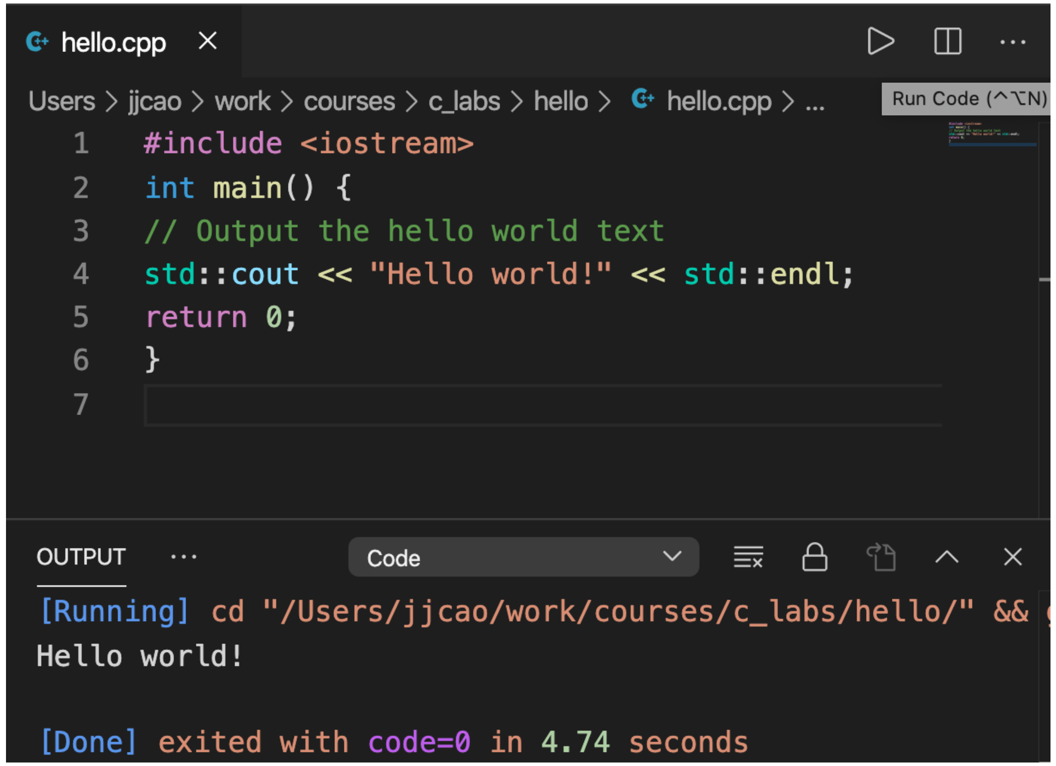
    ① cd "/Users/jjcao/work/courses/c_labs/hello/”

    ② g++ hello.cpp -o hello

    ③ "/Users/jjcao/work/courses/c_labs/hello/"hello

  ② Hello world!

  ③ [Done] …

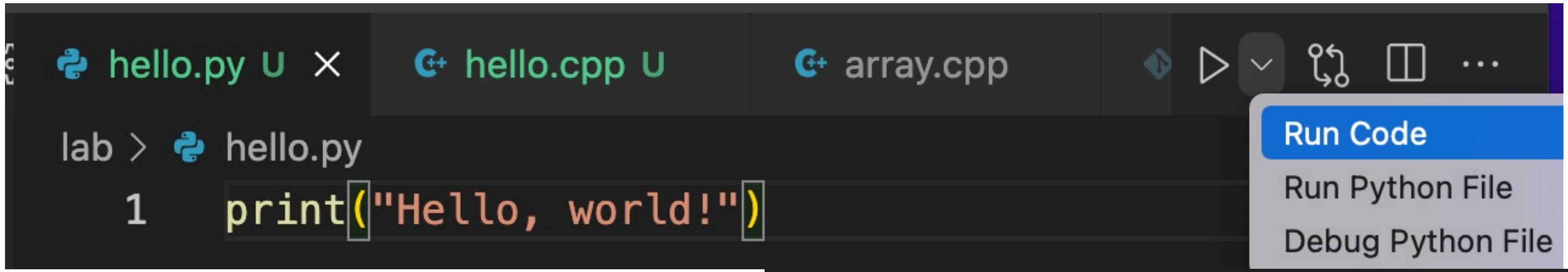

hello.cpp
Users > jjcao > work > courses > c_labs > hello > hello.cpp > ...
Run Code (^⌥N)
OUTPUT
Code

```cpp
#include <iostream>
int main() {
// Output the hello world text
std::cout << "Hello world!" << std::endl;
return 0;
}
```

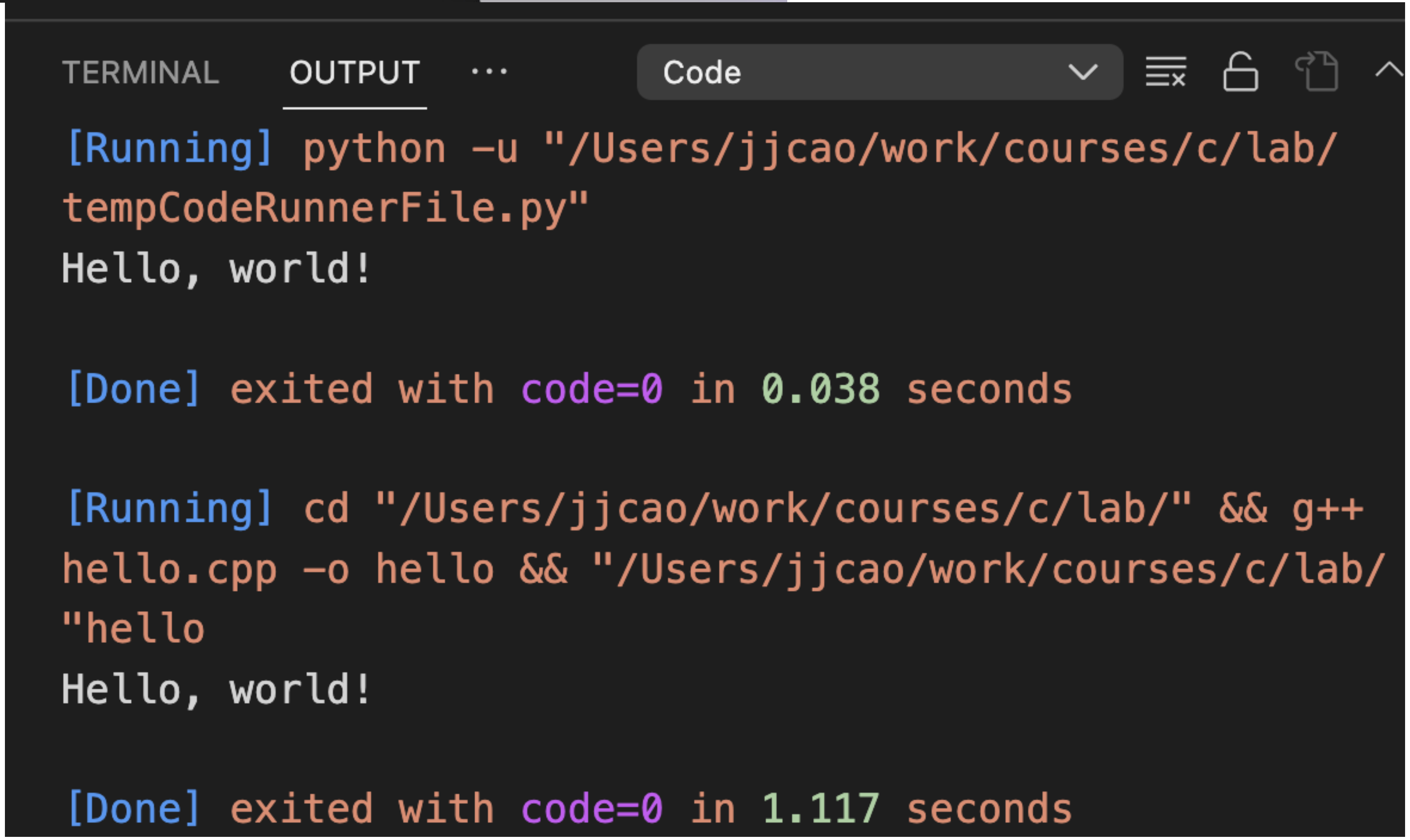
```
[Running] cd "/Users/jjcao/work/courses/c_labs/hello/" &&
Hello world!

[Done] exited with code=0 in 4.74 seconds
```

Congratulations!

Run it – python



```
print("Hello, world!")
```



```
[Running] python -u "/Users/jjcao/work/courses/c/lab/
tempCodeRunnerFile.py"
Hello, world!

[Done] exited with code=0 in 0.038 seconds

[Running] cd "/Users/jjcao/work/courses/c/lab/" && g++
hello.cpp -o hello && "/Users/jjcao/work/courses/c/lab/
"hello
Hello, world!

[Done] exited with code=0 in 1.117 seconds
```

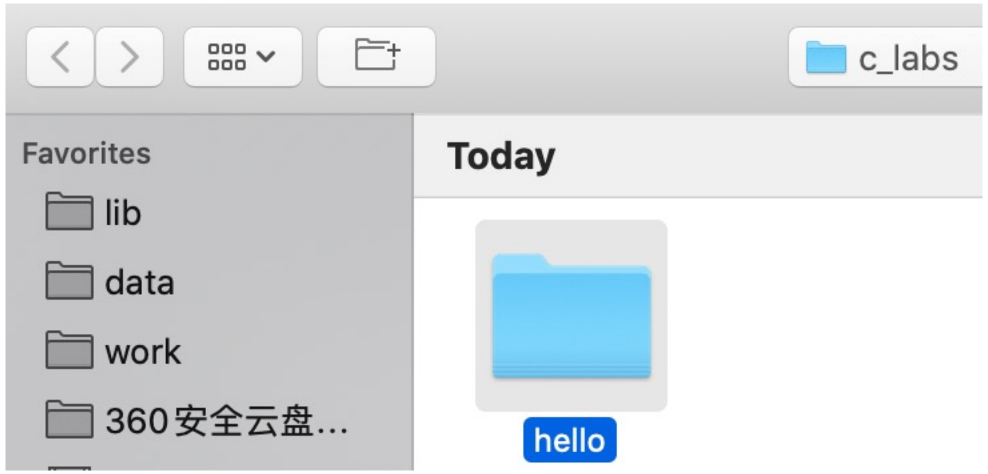
- 说说二者的不同？

interpreted vs. compiled language

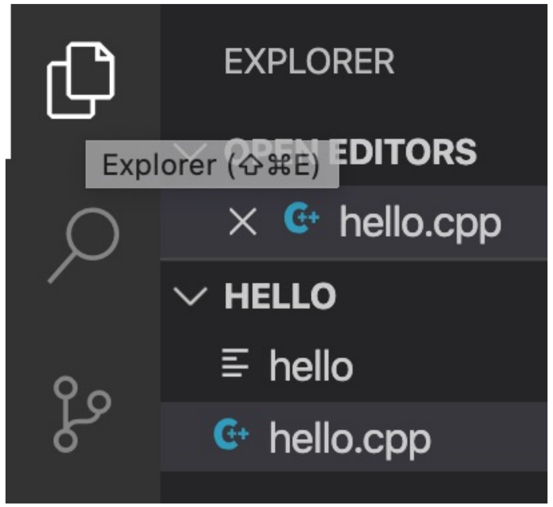
Python	C++
Interpreter解释器	Compiler编译器
directly executes statements in a scripting language without requiring them to have been assembled into machine language	generally transforms code written in a high-level language into a low-level language in order to create an executable program
Run a program in one step: run	Two steps: Compile / build, run
	Early detection of errors Faster program execution

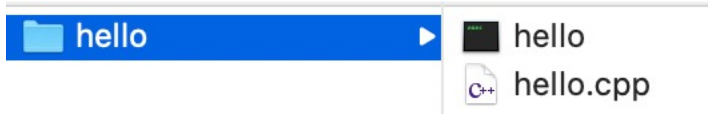
What do you have now

- File/open
  - Open the the project folder



- Click “Explorer”
  - Source file: hello.cpp
  - Executable file: hello or hello.exe

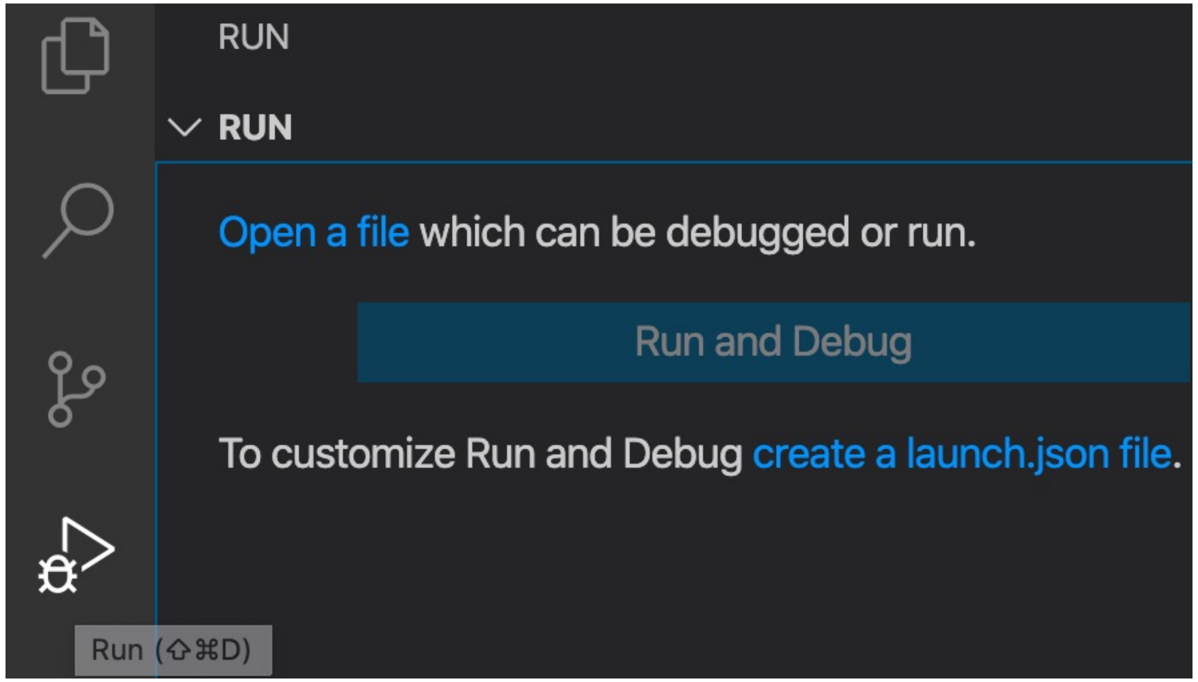




Questions?

Build and debug active file

- Create a launch.json file
  - C++ (GDB/LLDB)
    - clang++ - Build and debug active file


RUN

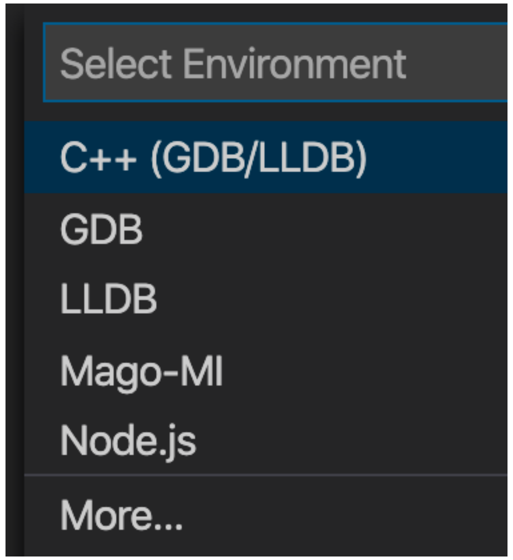
RUN

Open a file which can be debugged or run.

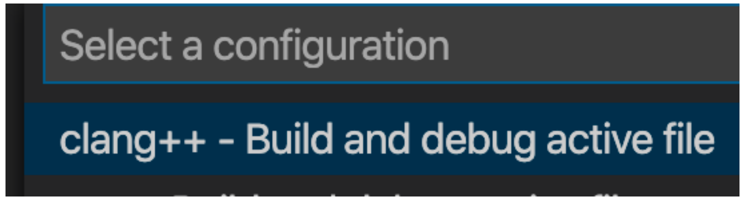
Run and Debug

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

Run (⇧⌘D)


Select Environment

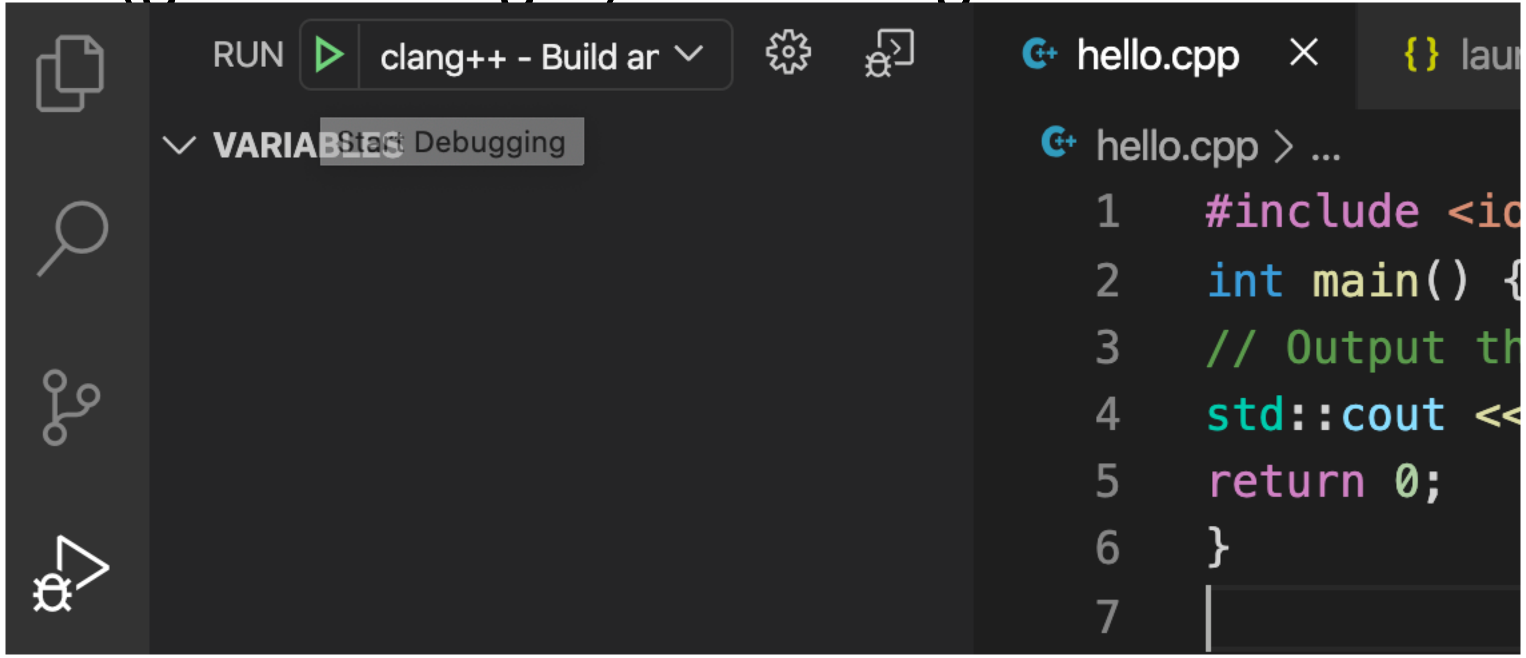
C++ (GDB/LLDB)
GDB
LLDB
Mago-MI
Node.js
More...


Select a configuration

clang++ - Build and debug active file

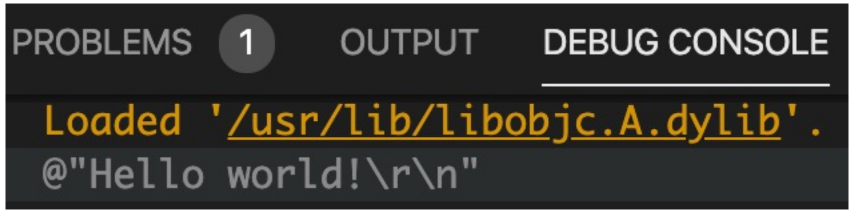
Run it – c++

- Another way without using "Code Runner" extension
- Run (green triangle) with clang++ - Build …



```
#include <ic
int main() {
// Output th
std::cout <<
return 0;
}
```

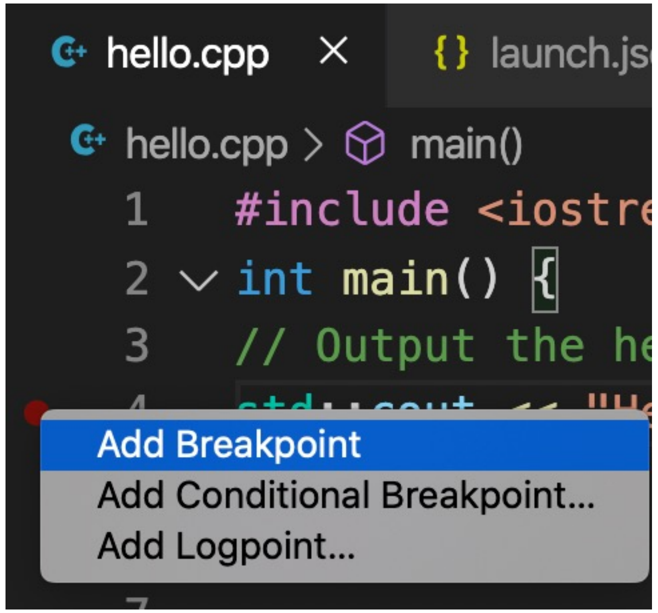
- Starting build...
- /usr/bin/clang++ -g /Users/jjcao/work/courses/c/lab/hello.cpp -o /Users/jjcao/work/courses/c/lab/bin/hello
- Build finished successfully.



```
Loaded '/usr/lib/libobjc.A.dylib'.
@"Hello world!\r\n"
```

Debug it

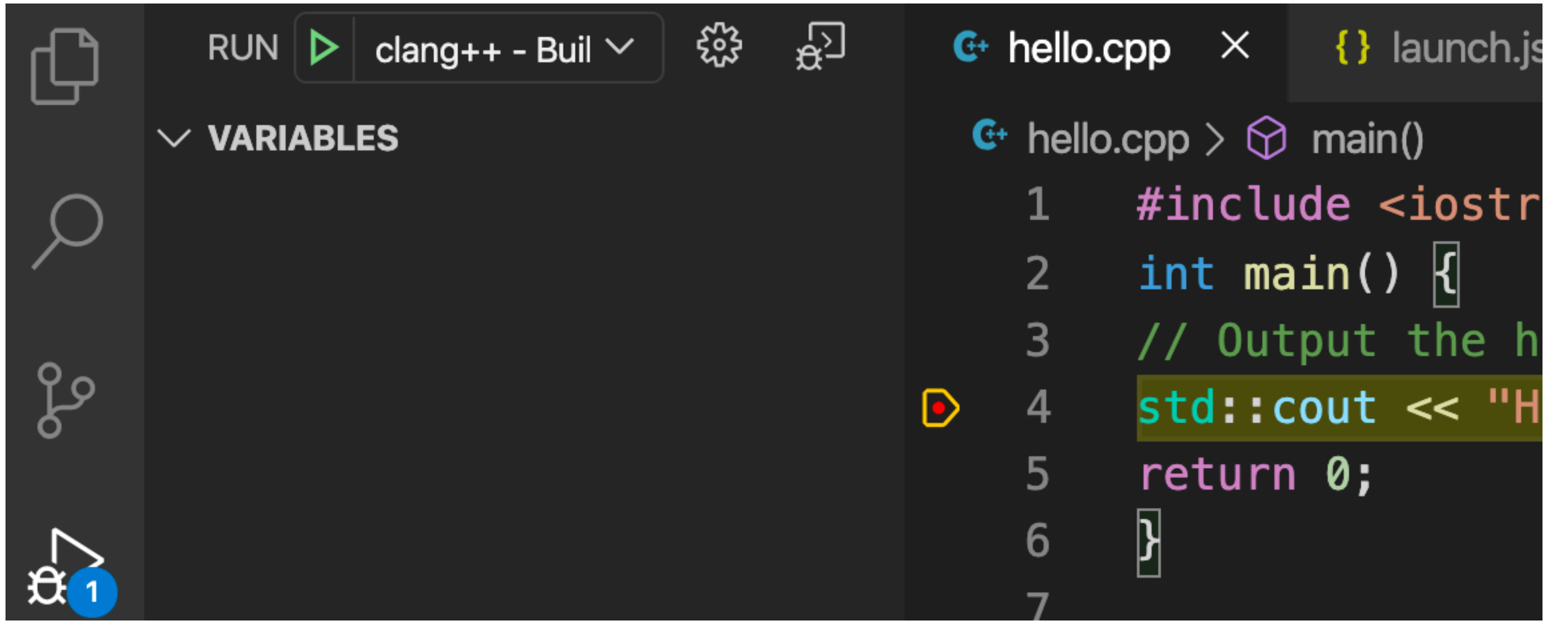
- Click a breakpoint
- Run (green triangle) with clang++ - Build …



hello.cpp × {} launch.js

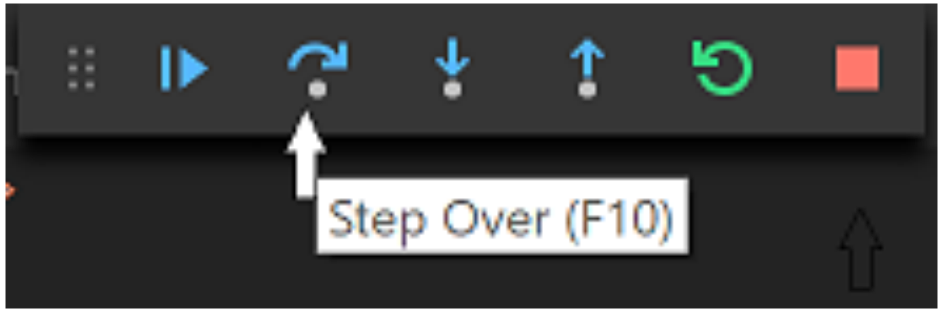
hello.cpp > main()

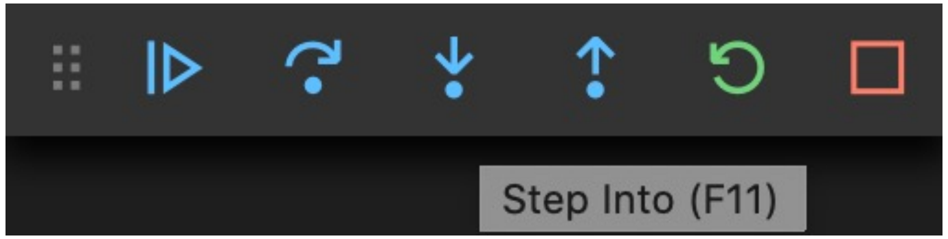
```
#include <iostre
int main() {
// Output the he
std::cout << "He
```



```
#include <iostr
int main() {
// Output the h
std::cout << "H
return 0;
}
```

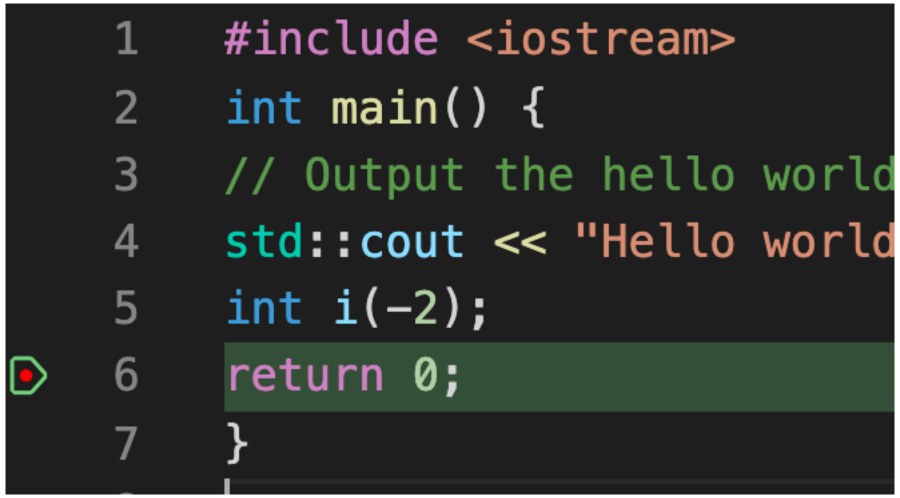
Step through the code





Add variables to the cpp

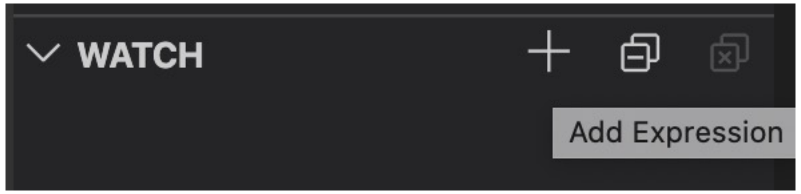
- Add
  - int i(-2);
- Add a breakpoint
- Debut the cpp
  - Then it stopped at line 6

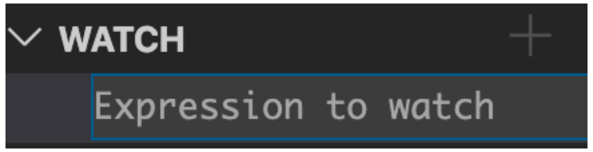


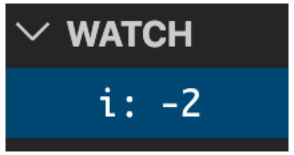
```
1  #include <iostream>
2  int main() {
3  // Output the hello world
4  std::cout << "Hello world
5  int i(-2);
6  return 0;
7  }
```

Set a watch

- Add a watch

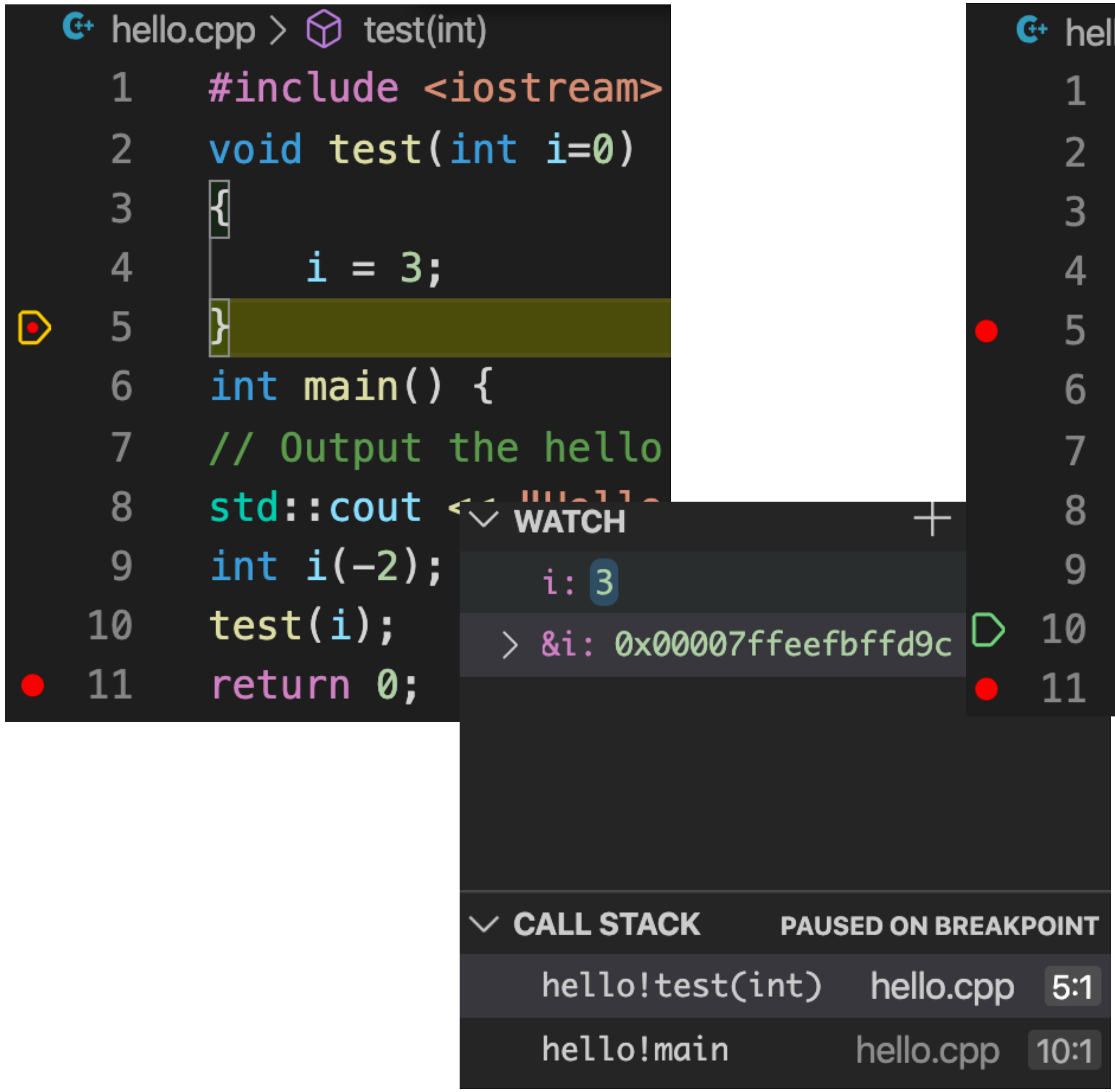






```
i: -2
```

Call Stack



hello.cpp > test(int)

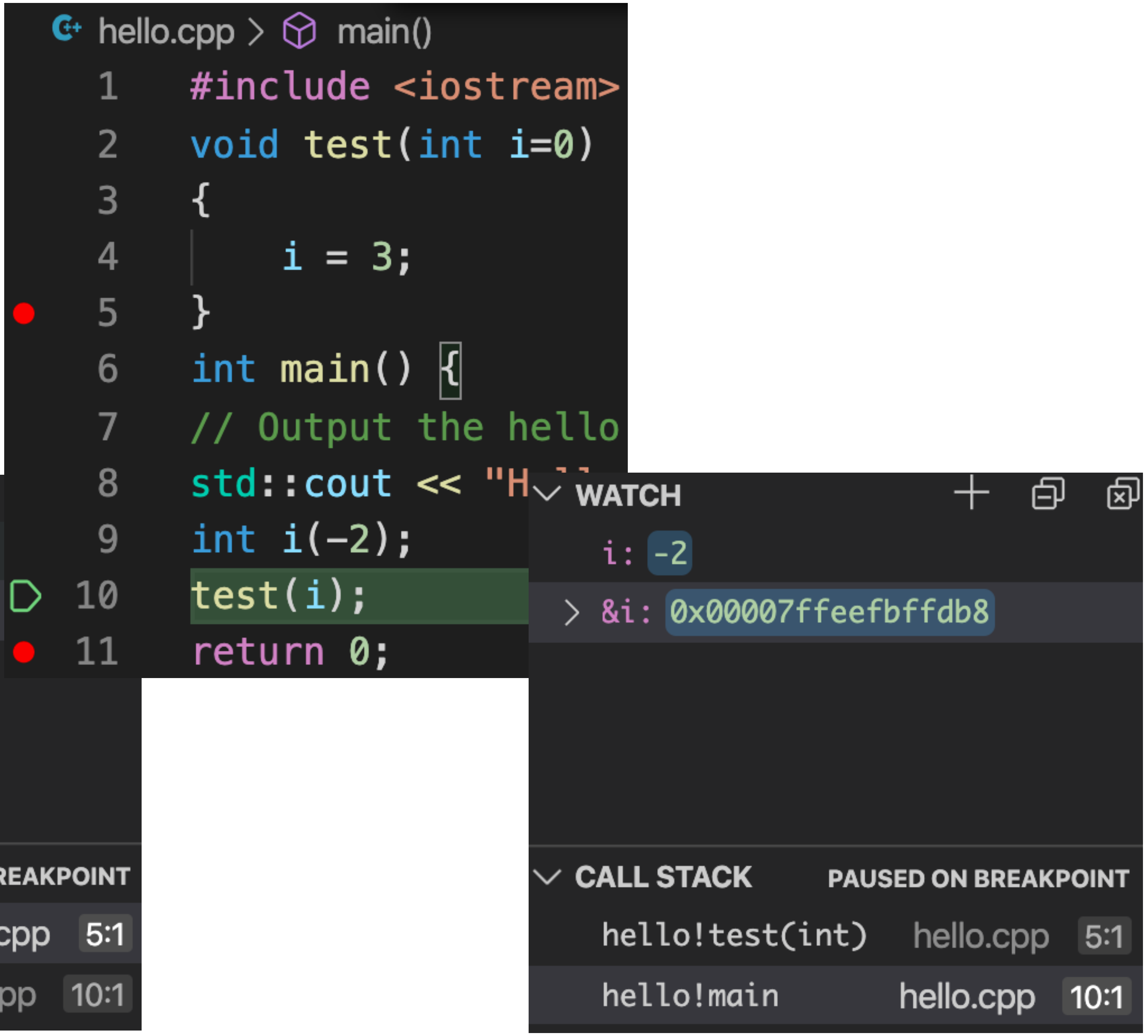
```cpp
#include <iostream>
void test(int i=0)
{
    i = 3;
}
int main() {
// Output the hello
std::cout <<
int i(-2);
test(i);
return 0;
```

WATCH

```
i: 3
> &i: 0x00007ffeefbffd9c
```

CALL STACK — PAUSED ON BREAKPOINT

```
hello!test(int)   hello.cpp   5:1
hello!main        hello.cpp   10:1
```



hello.cpp > main()

```cpp
#include <iostream>
void test(int i=0)
{
    i = 3;
}
int main() {
// Output the hello
std::cout << "H
int i(-2);
test(i);
return 0;
```

WATCH

```
i: -2
> &i: 0x00007ffeefbffdb8
```

CALL STACK — PAUSED ON BREAKPOINT

```
hello!test(int)   hello.cpp   5:1
hello!main        hello.cpp   10:1
```

Debugging your code

Six Stages of Debugging

1. That can't happen.
2. That doesn't happen on my machine.
3. That shouldn't happen.
4. Why does that happen?
5. Oh, I see.
6. How did that ever work?

Windows debugging with GDB

- Windows Debugging with MinGW64
- https://code.visualstudio.com/docs/cpp/config-mingw

Compile ?

Step 1: Modify Main

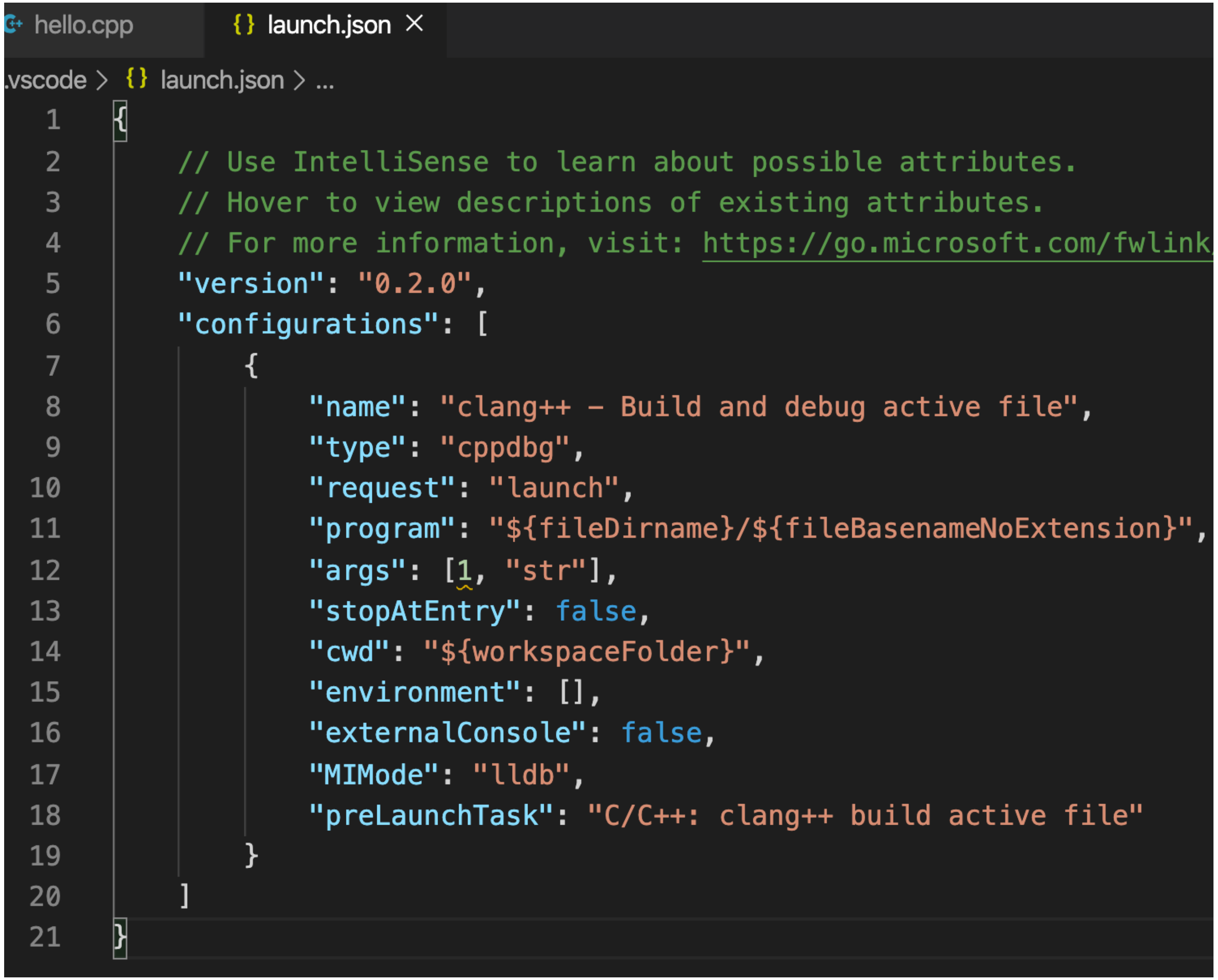
```
#include <iostream>
using namespace std;
int main(int argc, char** args)
{
// Notice I start from i=1 not 0 because the args[0] is reserved
//        for the name of this program.
          for(int i = 1; i < argc; i++)
          {
              cerr << i << "th argument is " << args[i] << "\n";
          }
}
```

Note: The relationship of argc and args:

1. args is **an array of char***
2. argc is the size of the array: args, which is determined when command line arguments are passed to the main() function. So after you change the size of args, argc is not updated automatically.

Launch.json

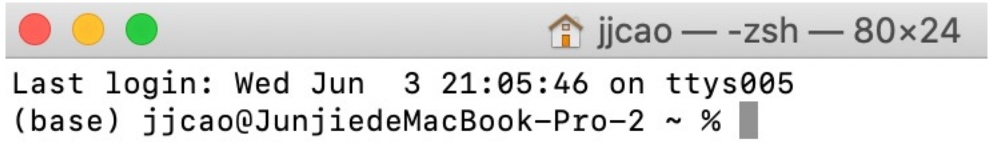
- “args”: [1, “str”]



```
{
    // Use IntelliSense to learn about possible attributes.
    // Hover to view descriptions of existing attributes.
    // For more information, visit: https://go.microsoft.com/fwlink
    "version": "0.2.0",
    "configurations": [
        {
            "name": "clang++ - Build and debug active file",
            "type": "cppdbg",
            "request": "launch",
            "program": "${fileDirname}/${fileBasenameNoExtension}",
            "args": [1, "str"],
            "stopAtEntry": false,
            "cwd": "${workspaceFolder}",
            "environment": [],
            "externalConsole": false,
            "MIMode": "lldb",
            "preLaunchTask": "C/C++: clang++ build active file"
        }
    ]
}
```

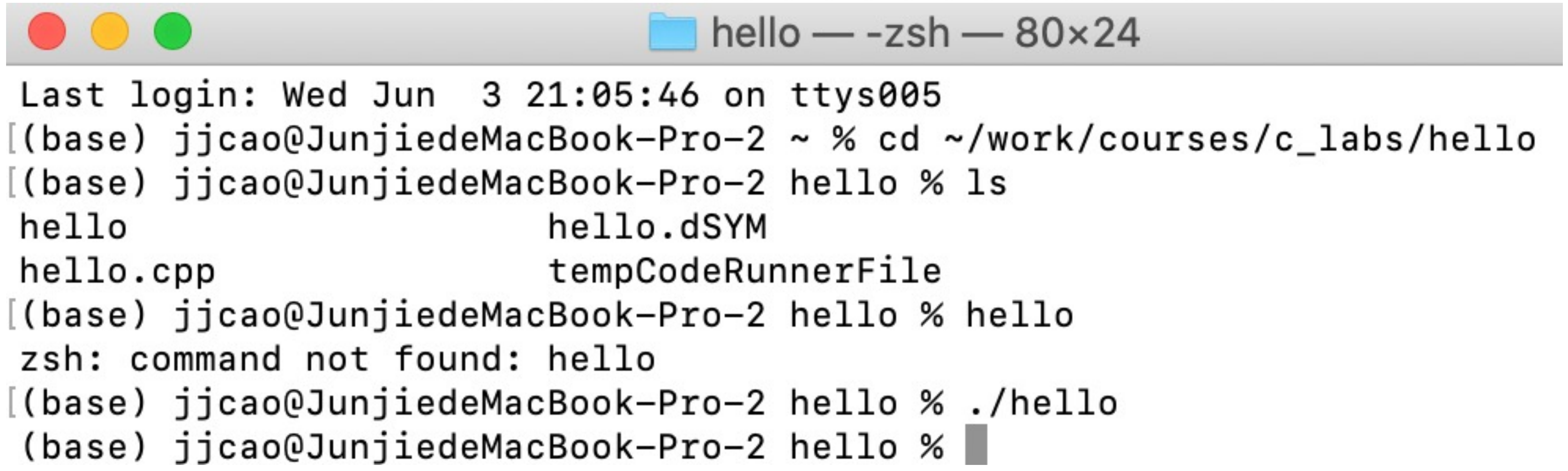
2. Command Line Argument命令行参数

1.Open a Terminal in Mac.



```
Last login: Wed Jun  3 21:05:46 on ttys005
(base) jjcao@JunjiedeMacBook-Pro-2 ~ %
```

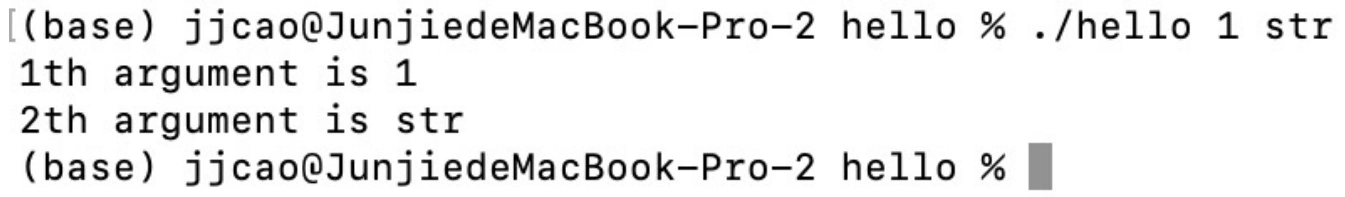
2. cd the project folder



```
Last login: Wed Jun  3 21:05:46 on ttys005
(base) jjcao@JunjiedeMacBook-Pro-2 ~ % cd ~/work/courses/c_labs/hello
(base) jjcao@JunjiedeMacBook-Pro-2 hello % ls
hello                       hello.dSYM
hello.cpp                   tempCodeRunnerFile
(base) jjcao@JunjiedeMacBook-Pro-2 hello % hello
zsh: command not found: hello
(base) jjcao@JunjiedeMacBook-Pro-2 hello % ./hello
(base) jjcao@JunjiedeMacBook-Pro-2 hello %
```

3.run the program: hello

1. hello
2. ./hello
3. ./hello 1 str



```
(base) jjcao@JunjiedeMacBook-Pro-2 hello % ./hello 1 str
1th argument is 1
2th argument is str
(base) jjcao@JunjiedeMacBook-Pro-2 hello %
```

Questions

User Input

- Let user input values to the program (line 6)

```
 1  #include <iostream>
 2  using namespace std;
 3
 4  int main() {
 5    int x;
 6    cin >> x;
 7
 8    cout << x / 3 << ' ' << x * 2;
 9
10    return 0;
11  }
```

- ctrl+z: cancel input from cin

iostream

- cin
- cout
- cerr
- clog
- Ordinarily, sys associates them with the console window.
- They can be redirected to files.

```
>>> num = input('Enter a number: ')
Enter a number: 10
>>> num
'10'
```

Thanks