

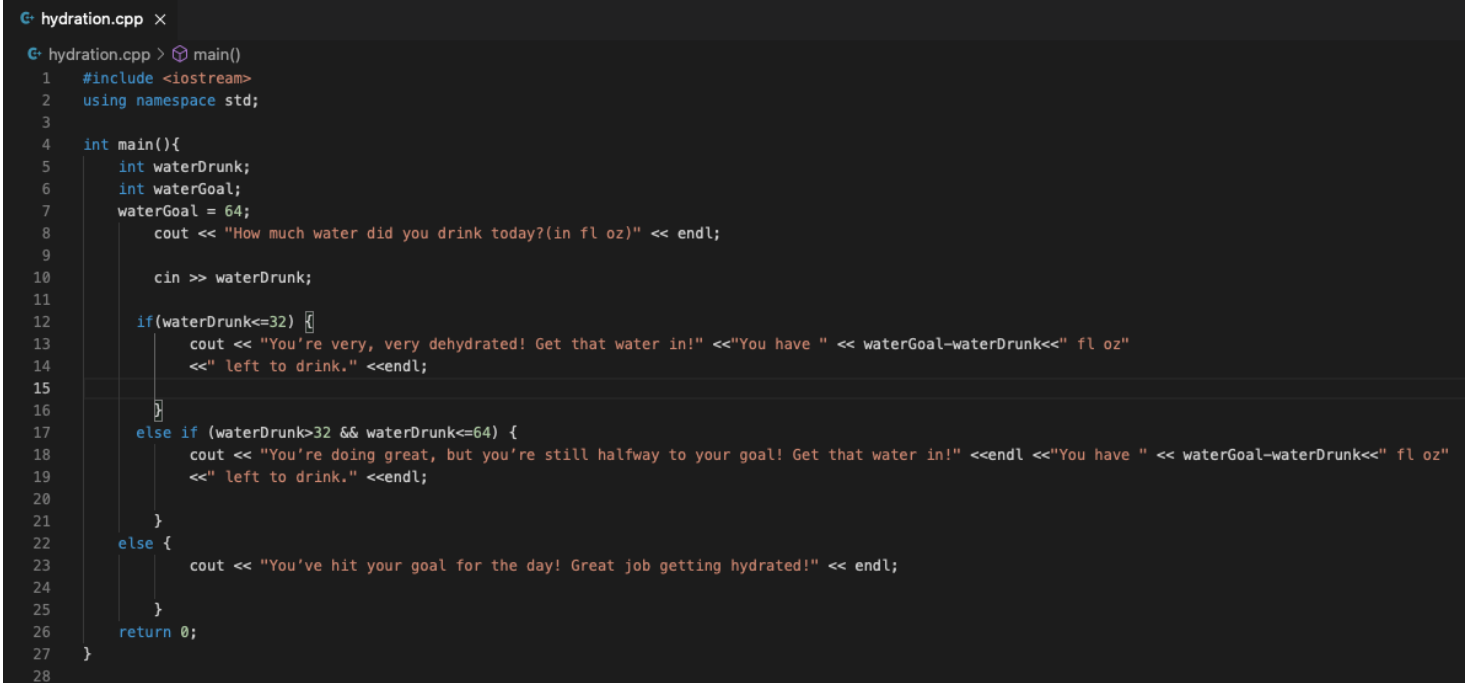
GUI Debugger for VS Code on CSEL

This guide is meant to help setup GNU Debugger (GDB), for students using VS Code on coding.csel.io or cs1300.csel.io. This guide will also help students who are using a local installation of VS Code on Linux.

For setting up VS Code on CSEL, please follow the CSEL VS Code Setup Guide. If you're using Linux, please follow the instructions detailed in the Linux VS Code Setup guide.

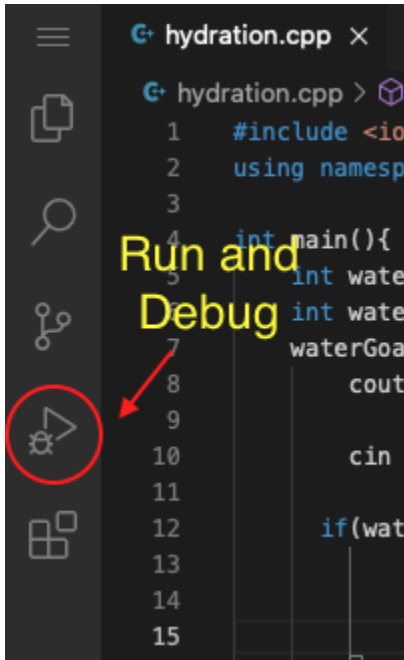
Step 1: Open VS Code using your CSEL account. Or if using Linux, open VSCode (you might have to search for it but you can also save it as a Favorite app to avoid having to do this in the future.) **The rest of the steps apply if you're using CSEL or Linux.**

Step 2: Let us debug the code for the **Hydration App** question from recitation 3.



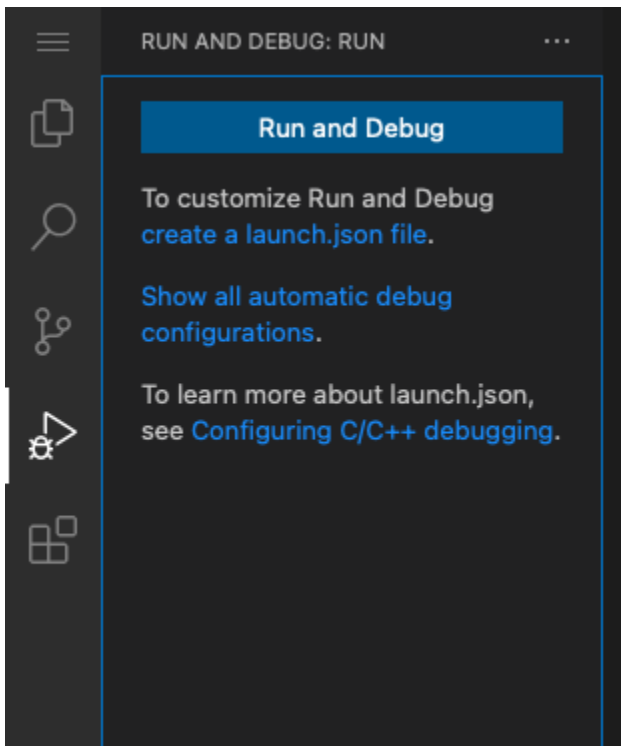
```
hydration.cpp x
hydration.cpp > main()
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5      int waterDrunk;
6      int waterGoal;
7      waterGoal = 64;
8      cout << "How much water did you drink today?(in fl oz)" << endl;
9
10     cin >> waterDrunk;
11
12     if(waterDrunk<=32) {
13         cout << "You're very, very dehydrated! Get that water in!" <<"You have " << waterGoal-waterDrunk<<" fl oz"
14         <<" left to drink." <<endl;
15     }
16
17     else if (waterDrunk>32 && waterDrunk<=64) {
18         cout << "You're doing great, but you're still halfway to your goal! Get that water in!" <<endl <<"You have " << waterGoal-waterDrunk<<" fl oz"
19         <<" left to drink." <<endl;
20     }
21
22     else {
23         cout << "You've hit your goal for the day! Great job getting hydrated!" << endl;
24     }
25
26     return 0;
27 }
28
```

Step 3: Click the “**Run and Debug**” icon on the **Activity Bar**.

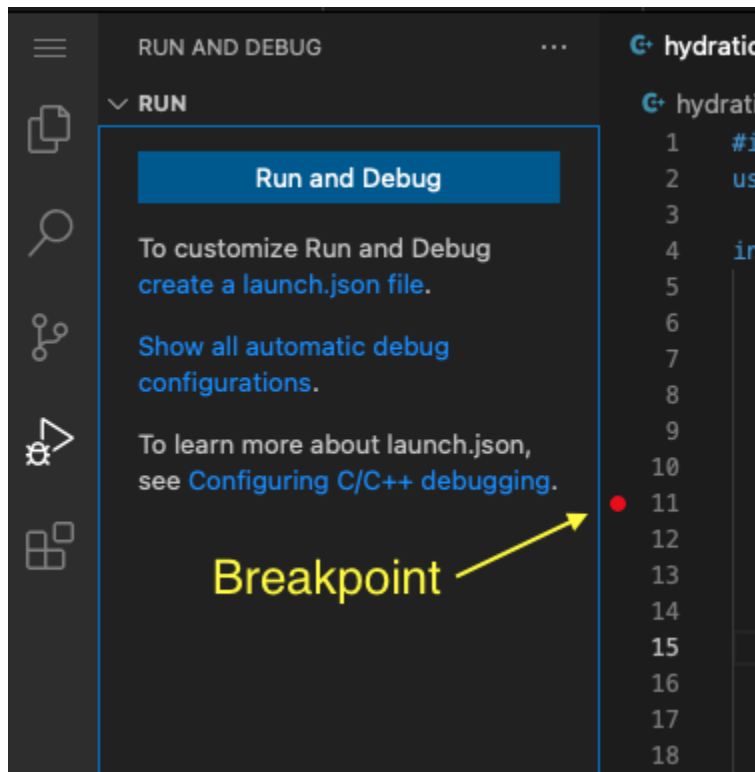


The Run view displays all information related to running and debugging and has a top bar with debugging commands and configuration settings.

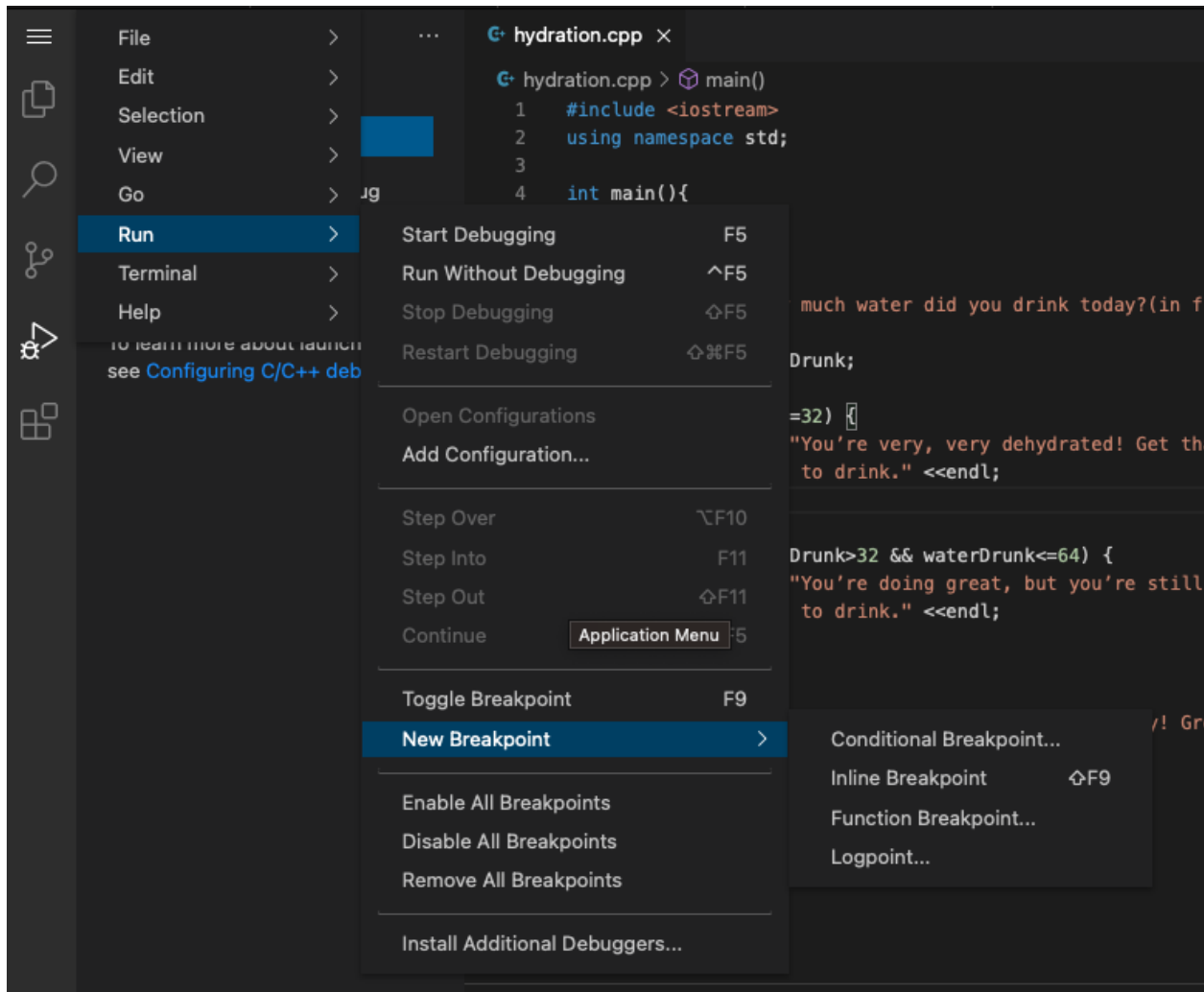
If running and debugging is not yet configured (no launch.json has been created) we see the Run start view.



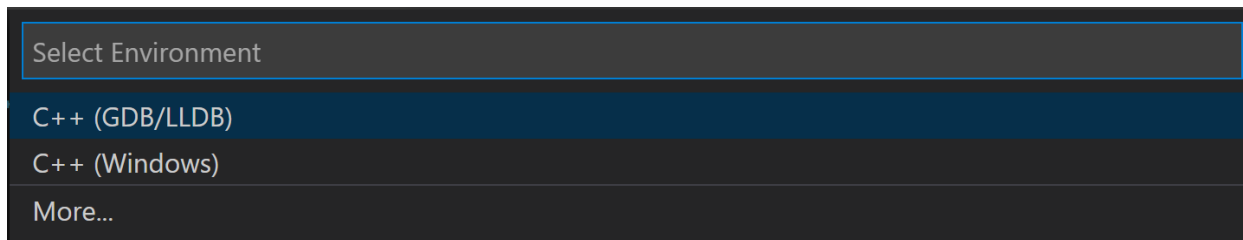
Step 4: To debug the program, we need to add a breakpoint. An easy way to do this is to click on the margin on the left of the line numbers like so, to create a red dot.

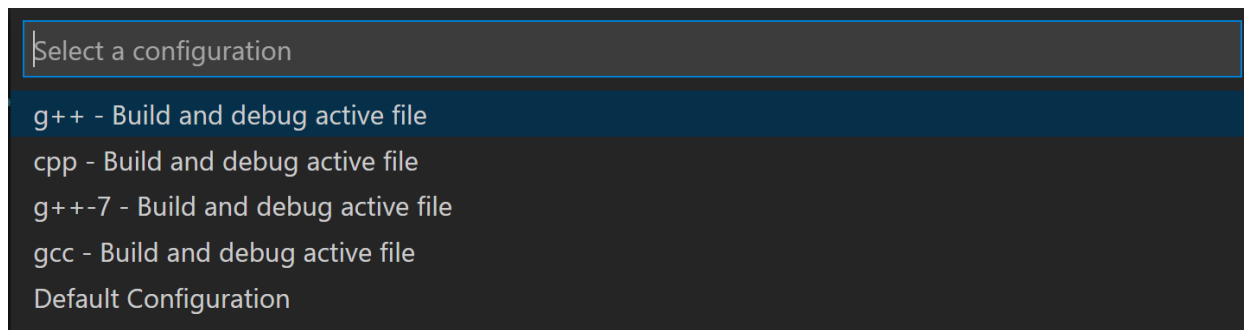


An alternative way is to go to the Run menu at the top, and select New Breakpoint → Inline Breakpoint. This adds a breakpoint to whatever line your cursor is currently on.

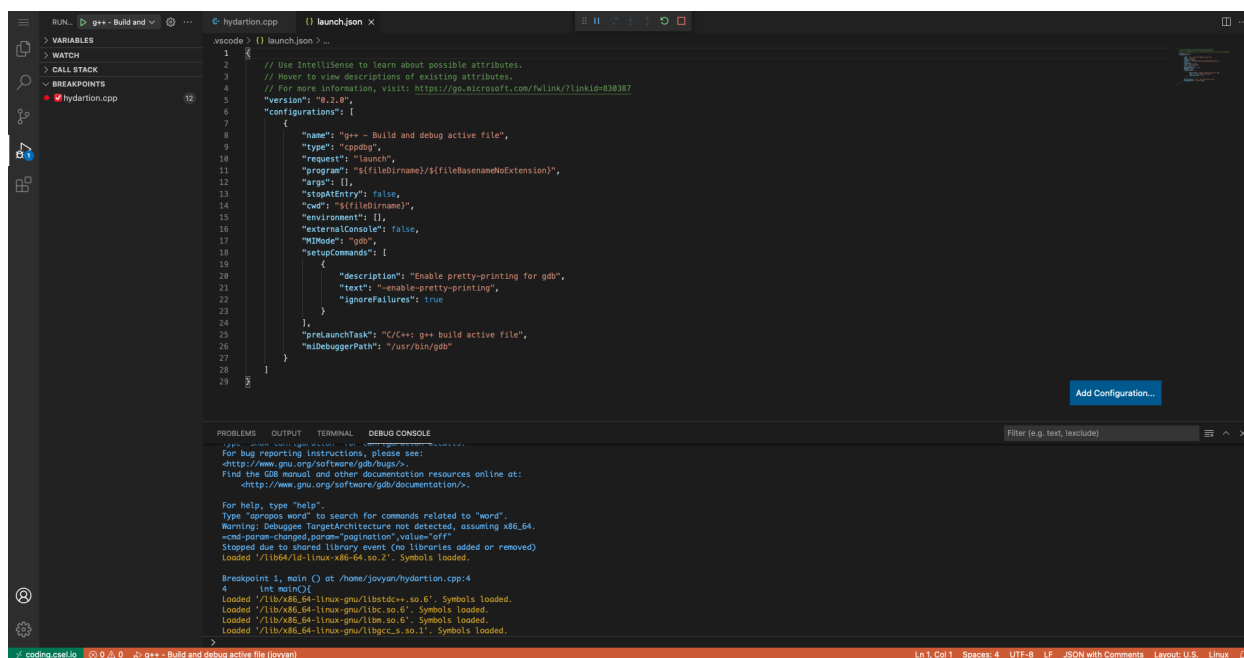


Step 5: Now that we have a breakpoint set, we can select the blue “Run and Debug” bar. Here, we are prompted to make a few selections: First, select **C++ (GDB/LLDB)** and then select **g++ - Build and debug active file** (you might see multiple options of this, select the first one). If you don’t see **C++ (GDB/LLDB)**, click **More...**, find the **C/C++ extension**, and install it.



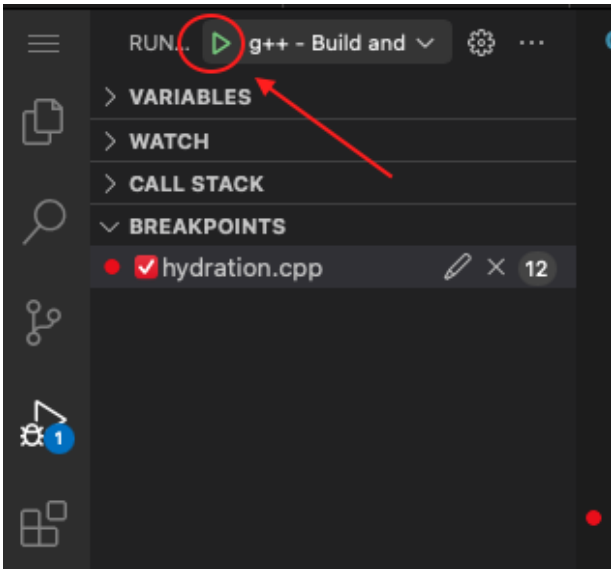


VSCode should now bring you to the file it has just created named **launch.json**, looking something like this:

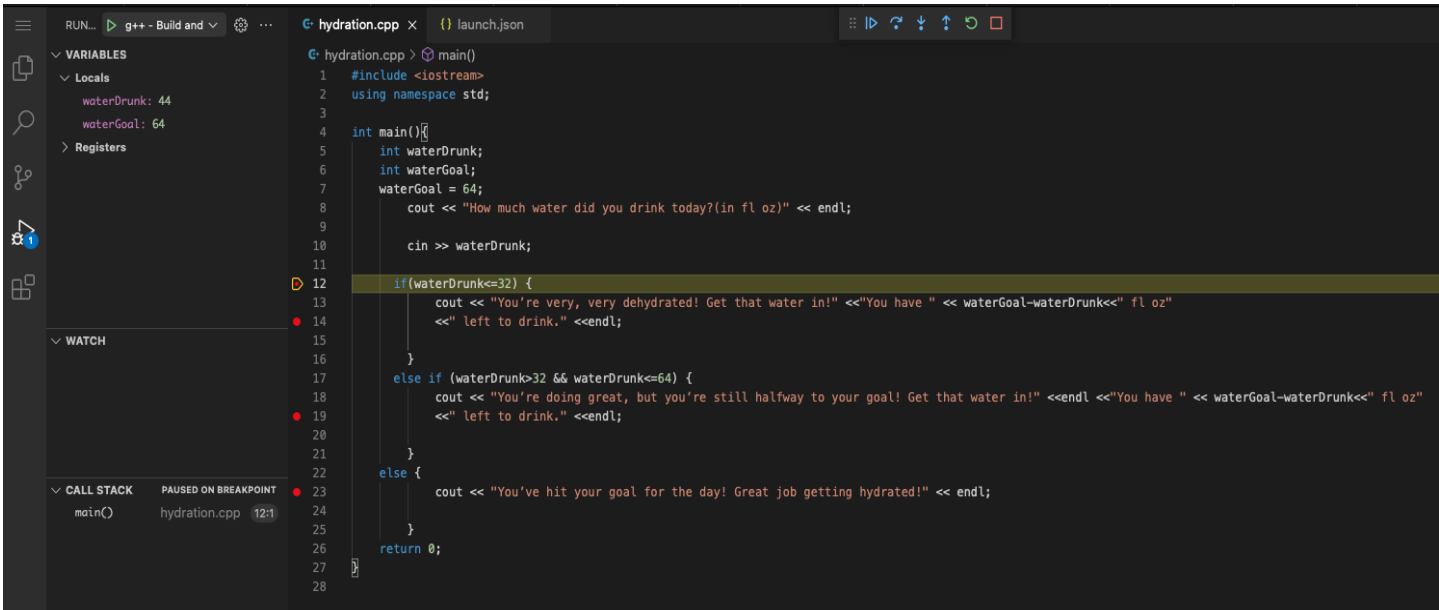


These are default launch configurations that VSCode has come up with to debug your file. Note: Sometimes It might start debugging your program as soon as the debugger is installed.

Step 6: To actually get started debugging, move to the .cpp file you'd like to debug, and click on the green Play button in the upper left corner of the debugging sidebar.



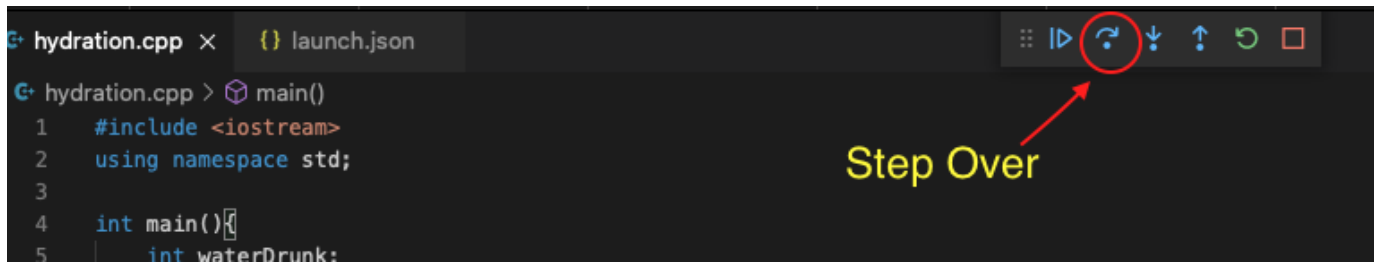
If you set a breakpoint and have launch.json created successfully, it should begin to run in debug mode. It should look something like this:



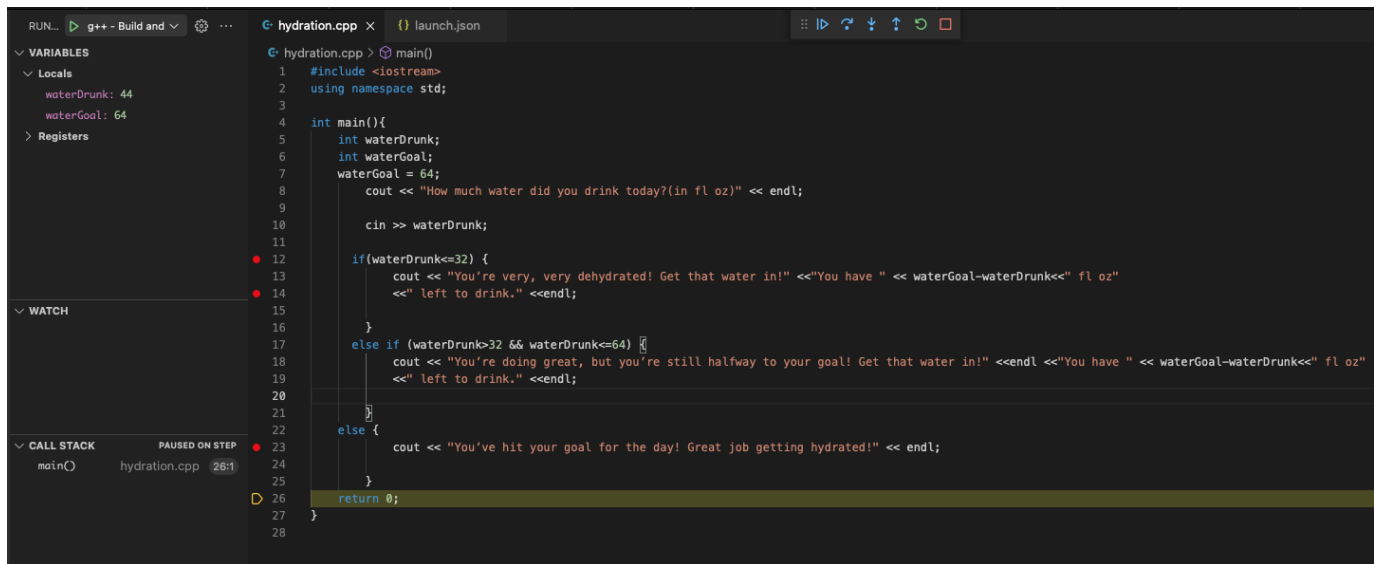
After taking the input (since our breakpoints are after the input) The debugger has run up to line 12 (and it hasn't executed line 12 yet). It is waiting for the user (you!) to tell it what to do next. There are a variety of common debugging commands we can use up at the top. Hover over each icon to see what they're called.

It is also worth noting that you can see the values of local variables in the VARIABLES pane on the left.

We can execute line 12 by clicking on the Step Over button. Before you click, take a look at the terminal - it should be empty!



After you click, notice that line 12 has been executed and the yellow highlighting moves to the next condition in our if-else case. Stepping over further will give us the following output in the terminal. If you want to run all over to the next breakpoint, you can also click the Continue button, instead of stepping over manually.



How much water did you drink today?(in fl oz)

44

You're doing great, but you're still halfway to your goal! Get that water in!

You have 20 fl oz left to drink.

You can finish execution of the program by clicking either Continue (the first icon) or Step Over again.

Other options for debugging include, Step Into and Step Out, these are useful for stepping into and out of the user defined functions and debugging them line by line.

Note: This guide is based on the official VS Code debugging guide available [here](#).