**Lab 2 C++ On Linux for HPC**

I don’t actually care what workflow you use for development, find something that works for you and go with it.

**Tools:**

VS code (you should have this), WSL 2 from the first lab.

Configuration: <https://code.visualstudio.com/docs/cpp/config-wsl>

Basically: Install Ubuntu (or some linux flavour).

Update it

Create a directory for your code

sudo apt-get install build-essential gdb

(that’s the gcc compiler and debugger)

From the WSL terminal go to the directory you want to run code from type “ code . “ (

code .

That brings up VS code and connects you to it.

Create a hello world.cpp file

Install any C++ extensions that you want (at the very least the MS C/C++ one)

|  |
| --- |
| #include <iostream>  #include <vector>  #include <string>  using namespace std;  int main()  {  vector<string> msg {"Hello", "C++", "World", "from", "VS Code", "and the C++ extension!"};  for (const string& word : msg)  {  cout << word << " ";  }  cout << endl;  } |

Save.

Run: To do that select the g++ build and debug active file option (I think)

Now for C

Make a new program (factorial.cpp) this is actually a C program but it’s the same compiler and you’ll find we interchange the two a bit thanks to how thing work in this class

# include <stdio.h>

int main()

{

int i, num, j;

printf ("Enter the number: ");

scanf ("%d", &num );

for (i=1; i<num; i++)

j=j\*i;

printf("The factorial of %d is %d\n",num,j);

}

Set a breakpoint (F9)

The play button can either be just run mode or debug mode select the debug mode

# Lab for Today:

As a supplement if you aren’t familiar with C++ there is a simple guide on C++ programming attached as well

1. Create a 2 dimensional array of floating point values

Here is an array:  
 {0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9},

{ 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9 },

{ 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9 },

{ 3.0, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9 },

{ 4.0, 4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9 },

{ 5.0, 5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9 },

{ 6.0, 6.1, 6.2, 6.3, 6.4, 6.5, 6.6, 6.7, 6.8, 6.9 },

{ 7.0, 7.1, 7.2, 7.3, 7.4, 7.5, 7.6, 7.7, 7.8, 7.9 },

{ 8.0, 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.7, 8.8, 8.9 },

{ 9.0, 9.1, 9.2, 9.3, 9.4, 9.5, 9.6, 9.7, 9.8, 9.9 },

(Feel free to copy paste that)

Sum up all of the values in the array and print the result in the form (do this in main)

The Sum is: 495 (this might not be the right value, I’ve changed the lab over the years)

Test the following errors in your code – in an inner loop have an array out of bounds error (so try and sum up for example for int j = 0, j < 11, j++ - array[i][j] and see what error it creates, and learn to debug.

Program runs every time

450

When change to column=1 495

For 2 and 3 you can hard code the dimensions of the array to 10 x10

1. Pass the array to a function which can calculate the sum of the diagonal and return the result, print the result from main.
2. Pass the array by reference to a void function which will square all of the values in the original array. Recalculate the sum and diagonal.