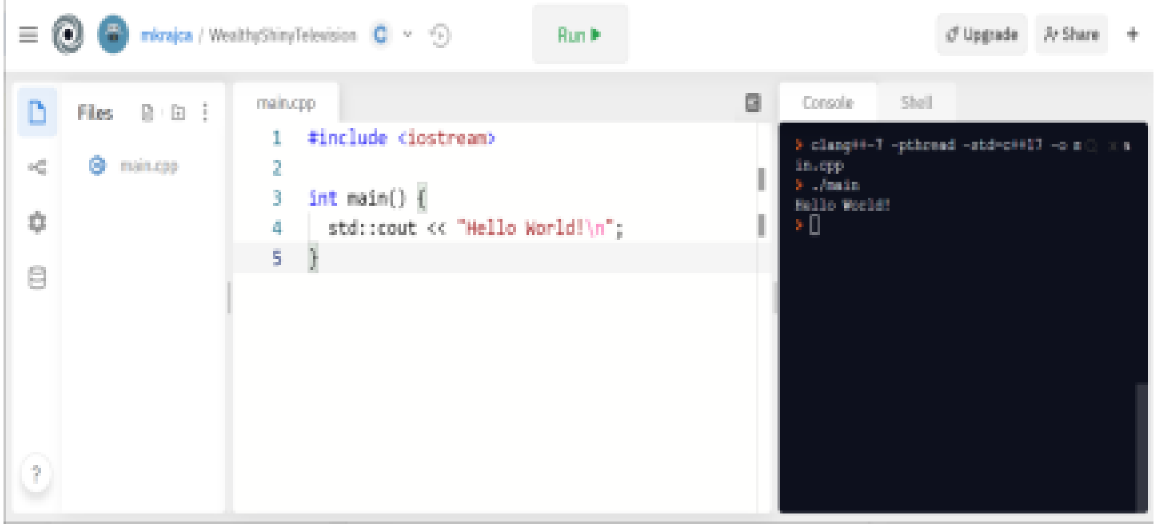
LAB 1

Name____Christian Morgan___________________________________

Objectives:

- Setup compiler environment and become familiar with it
- Learn basic debugging and compiling

1) From the class handout notes choose one of the environments (IDE's) and set them up. (visual studio, JetBrains Clion (needs education license, see notes on obtaining one for free), on linux use g++,….) If you don't have one available for your OS, then get familiar with an environment such as repl. https://repl.it/languages/cpp
Note you will have to create a repl account to use all the features. It is free.
Some Screen shots are used from repl since it is available to all operating systems and environments.



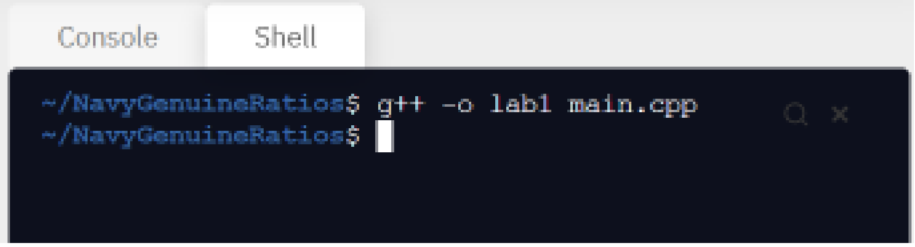
```cpp
#include <iostream>

int main() {
  std::cout << "Hello World!\n";
}
```

```
> clang++-7 -pthread -std=c++17 -o main ma
in.cpp
> ./main
Hello World!
> 
```

Most IDE's will have some sort of a rum button. On repl it is the center of the screen. Click it and see if your program compiles. Output should be in the console window. Almost all IDE's will create some sort of basic code when you create a project. If you don't have one type in the code that was in the repl screenshot.

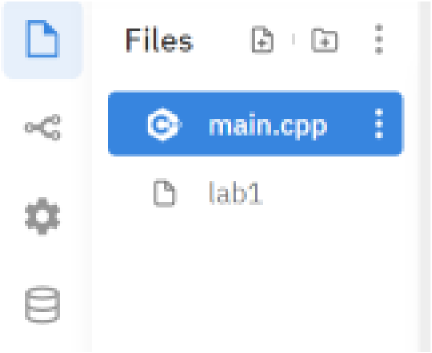
In repl we can use most Linux commands to compile including G++.

In the shell type:

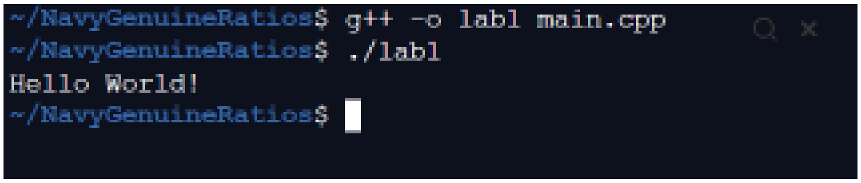


```
~/NavyGenuineRatios$ g++ -o lab1 main.cpp
~/NavyGenuineRatios$ 
```

The right-hand side should now have a new file:



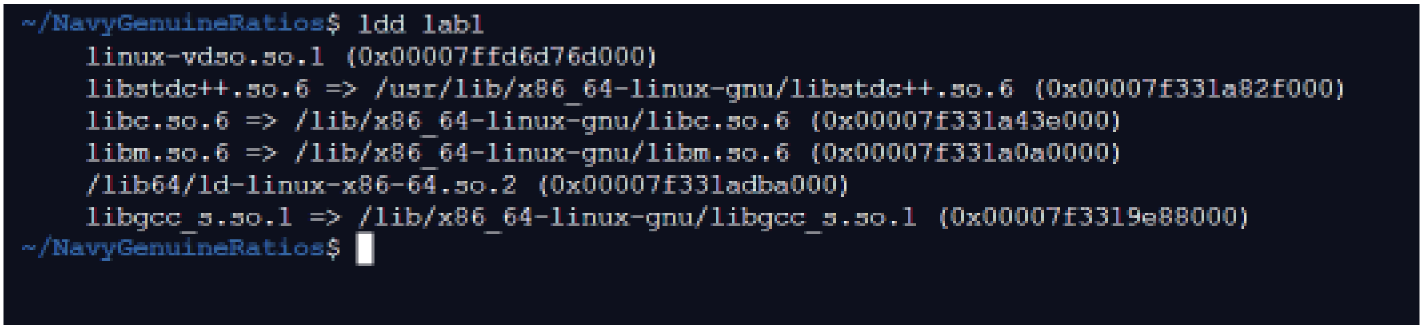
G++ is the c++ compiler. The -o tag tells it to create an executable named lab1. Note in Unix you don't have to use .exe extension. We can run the executable by typing in Repl by default use another c+= compiler called "clang"



```
~/NavyGenuineRatios$ g++ -o lab1 main.cpp
~/NavyGenuineRatios$ ./lab1
Hello World!
~/NavyGenuineRatios$
```

./lab1

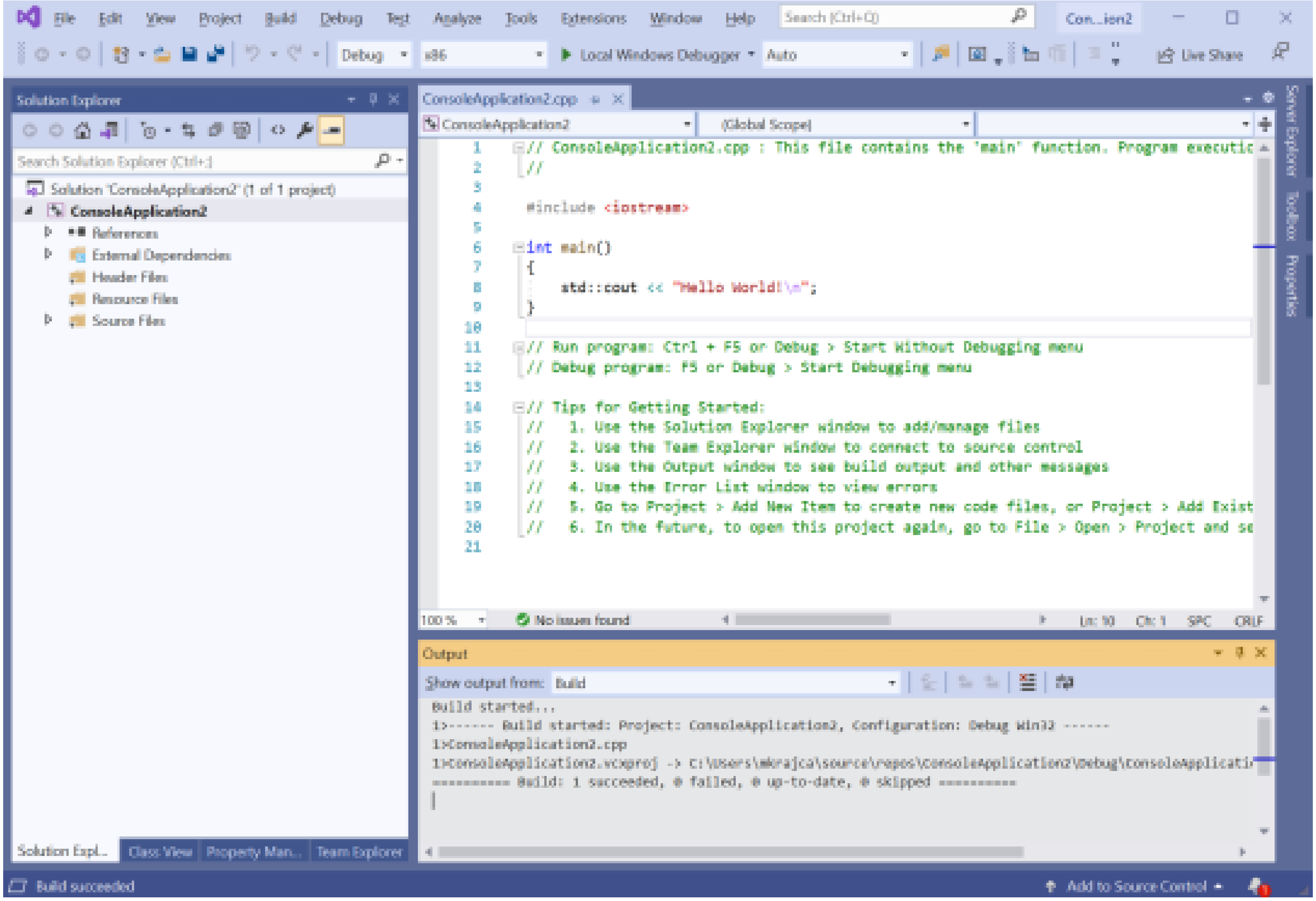
We can use other Unix/Linux commands to such as "ldd" to see what libraries were compiled in to out exectable.



```
~/NavyGenuineRatios$ ldd lab1
        linux-vdso.so.1 (0x00007ffd6d76d000)
        libstdc++.so.6 => /usr/lib/x86_64-linux-gnu/libstdc++.so.6 (0x00007f331a82f000)
        libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f331a43e000)
        libm.so.6 => /lib/x86_64-linux-gnu/libm.so.6 (0x00007f331a0a0000)
        /lib64/ld-linux-x86-64.so.2 (0x00007f331adba000)
        libgcc_s.so.1 => /lib/x86_64-linux-gnu/libgcc_s.so.1 (0x00007f3319e88000)
~/NavyGenuineRatios$
```

Most of these commands for debugging are not available on windows using visual studio or Clion (at least not easily)

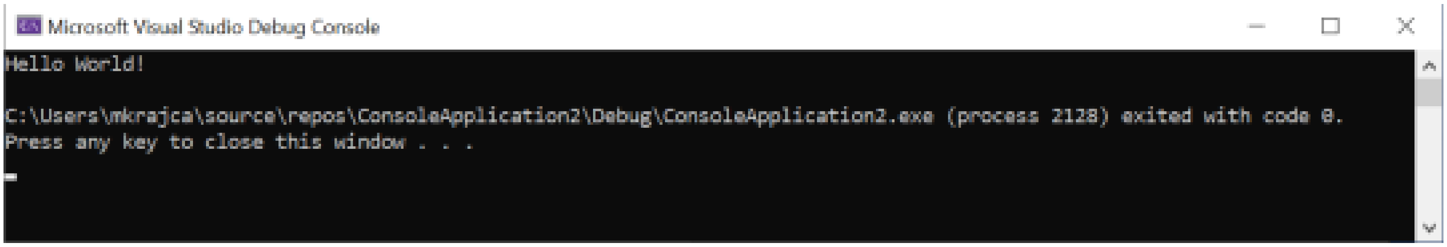
Visual studio:



```
// ConsoleApplication2.cpp : This file contains the 'main' function. Program executio
//

#include <iostream>

int main()
{
    std::cout << "Hello World!\n";
}

// Run program: Ctrl + F5 or Debug > Start Without Debugging menu
// Debug program: F5 or Debug > Start Debugging menu

// Tips for Getting Started:
//   1. Use the Solution Explorer window to add/manage files
//   2. Use the Team Explorer window to connect to source control
//   3. Use the Output window to see build output and other messages
//   4. Use the Error List window to view errors
//   5. Go to Project > Add New Item to create new code files, or Project > Add Exist
//   6. In the future, to open this project again, go to File > Open > Project and se

```

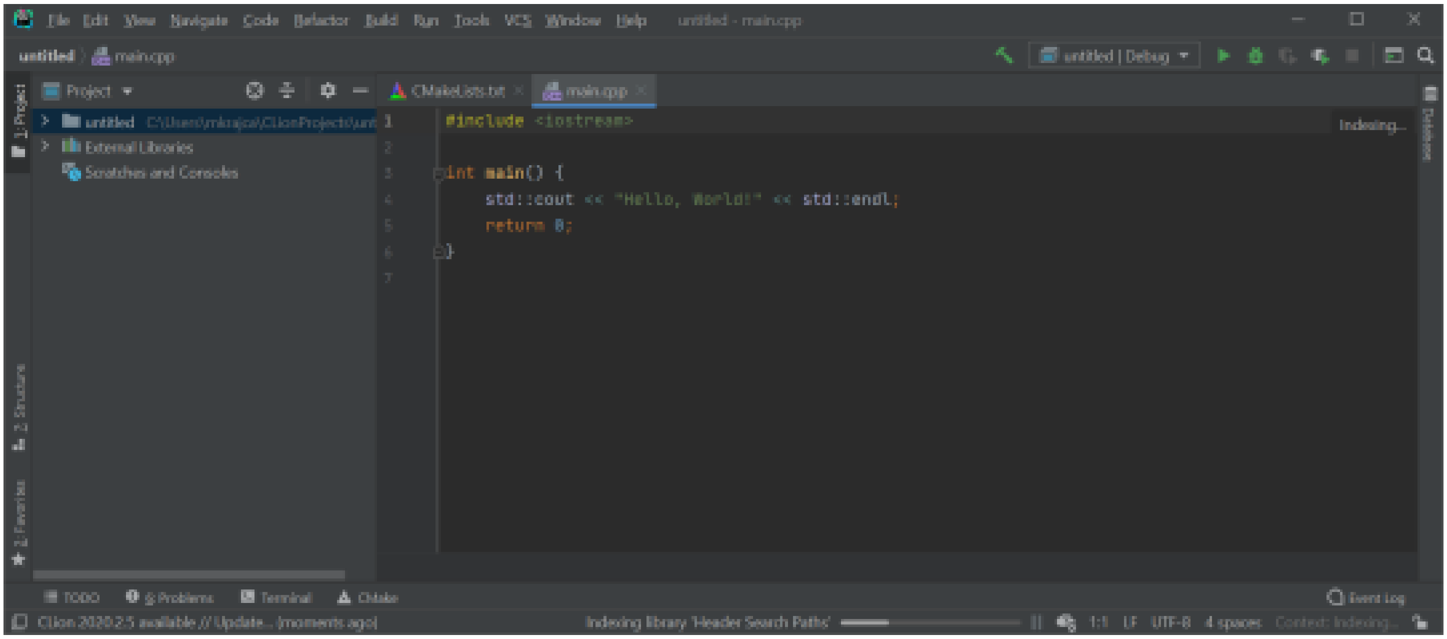
```
Build started...
1>------ Build started: Project: ConsoleApplication2, Configuration: Debug Win32 ------
1>ConsoleApplication2.cpp
1>ConsoleApplication2.vcxproj -> C:\Users\mkrajca\source\repos\ConsoleApplication2\Debug\ConsoleApplicati
========== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped ==========
```

Press CTRL + F5 for output window.



```
Hello World!

C:\Users\mkrajca\source\repos\ConsoleApplication2\Debug\ConsoleApplication2.exe (process 2128) exited with code 0.
Press any key to close this window . . .
```

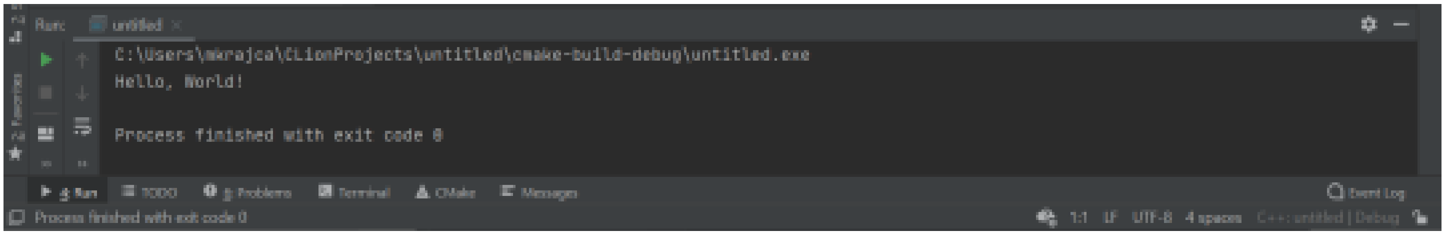
JetBrains Clion:



File Edit View Navigate Code Refactor Build Run Tools VCS Window Help untitled - main.cpp
untitled main.cpp
Project
CMakeLists.txt
main.cpp
untitled C:\Users\mkrajca\CLionProjects\unt
External Libraries
Scratches and Consoles
Indexing...
untitled | Debug
TODO
6: Problems
Terminal
CMake
Event Log
CLion 2020.2.5 available // Update... (moments ago)
Indexing library 'Header Search Paths'
1:1 LF UTF-8 4 spaces Context: Indexing...

```
#include <iostream>

int main() {
    std::cout << "Hello, World!" << std::endl;
    return 0;
}

```

Click the ▶ in the menu bar



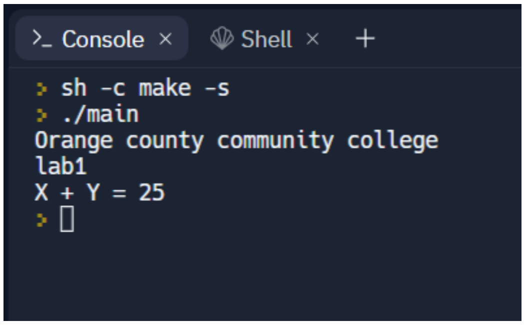
```
C:\Users\mkrajca\CLionProjects\untitled\cmake-build-debug\untitled.exe
Hello, World!

Process finished with exit code 0
```

Process finished with exit code 0

2) After you get familiar with the basic environment. Type in the following code.

```
#include <iostream>
using namespace std;
int main() {
 cout << "Orange county community college" << endl;
cout<<"lab1"<<endl;
 int x; // integer variable
 int y; // integer variable;
 x = 20;
 y = 5;
 cout<<"X + Y = " << x + y<<endl;

 return 0;
}
```

Run the program. (compile and run)
What is the output of the program? It printed Orange county community college and x + y = 25.



```
> sh -c make -s
> ./main
Orange county community college
lab1
X + Y = 25
> 
```

Delete the semicolon on line 8. Where x=20

```
#include <iostream>
using namespace std;
int main() {
 cout << "Orange county community college" << endl;
cout<<"lab1"<<endl;
 int x; // integer variable
 int y; // integer variable;
 x = 20
 y = 5;
 cout<<"X + Y = " << x + y<<endl;

 return 0;
}
```

Try to compile the code again.

Did it compile? If not what kind of error did you get? No, I got an exit status 2 error. They said error expected after expression x = 20.
Put back the semicolon and try to compile again.

Did it compile? Yes

Sometimes a program will compile but does not have the correct output. Most IDE's will have some sort of a trace feature or debug mode. However, if you are working on projects on Unix Linux that are remote (such as working on a project in Amazon cloud, or most cloud services there is not IDE available. Also, most embedded microcontrollers that have specialized CPUs the only way to debug is on the printout statements or using some other hardware such as Jtags (beyond the scope of this class)
Debugging manually is the only way to trace the program. We use "cout" to see the content of variables at runtime.

Modify the above program as follows:

```
#include <iostream>
using namespace std;
int main() {
 cout << "Orange county community college" << endl;
 cout<<"lab1"<<endl;
 int x; // integer variable
 int y; // integer variable;
 x = 20;
 cout<<"content of X right after initialization: " << x << endl;
y = 5;
 cout<<"X + Y = " << x + y<<endl;
 cout<<"content of y right before program exit: " << y << endl;

 return 0;
}
```

What kind of output did you get? Were you able to see the values of x and y during the program execution? Yes
Comment out the debugging "cout" statements"

```
#include <iostream>
using namespace std;
int main() {
 cout << "Orange county community college" << endl;
cout<<"lab1"<<endl;
 int x; // integer variable
 int y; // integer variable
 x = 20;
 //cout<<"content of X right after initialization: " << x << endl;   y
= 5;
 cout<<"X + Y = " << x + y<<endl;
 //cout<<"content of y right before program exit: " << y << endl;

 return 0;
}
```

Run the program. Did the contents of x and y display during the run of the program? No

Create a banner for your program. You should have a banner on your labs when you hand them in.
The banner should print out the Your name, class, lab number(or assignment) and the date.
Example:

```
#include <iostream>
using namespace std;
int main() {
 cout<<"---------------------------------"<<endl;
 cout << "Miroslav Krajca" << endl;
 cout<< "Computer Science 1"<<endl;
 cout<< "Lab1"<< endl;
 cout<< "January 27, 2021" << endl;
 cout<<"---------------------------------"<<endl;
 int x; // integer variable
 int y; // integer variable;
 x = 20;
 //cout<<"content of X right after initialization: " << x << endl;   y
= 5;
 cout<<"X + Y = " << x + y<<endl;
 //cout<<"content of y right before program exit: " << y << endl;

 return 0;
}
```

Paste the final code into your lab and turn it in.

```
#include <iostream>
using namespace std;
int main() {
  cout<<"----------------------"<<endl;
  cout<<"Christian Morgan"<<endl;
  cout<<"Computer Science 1"<<endl;
  cout << "Orange County Community College" << endl;
  cout<<"lab1"<< endl;
  cout<<"January 20, 2023"<<endl;
  cout<<"----------------------"<<endl;
  int x; // integer variable
  int y; // integer variable
  x = 20;
  //cout<<"content of X right after initialization: " << x << endl;
  y = 5;
  //cout<<"X + Y = " << x + y<< endl;

  return 0;
}
```