School of Computer Science
Faculty of Science

COMP-2650: Computer Architecture I: Digital Design
Fall 2020

Lab#	Date	Title	Due Date	Grade Release Date
Lab 01	Sept 21-23, 2020	**L01: Programming Environment Setup**	Sept. 30, 2020 Wednesday Midnight AoE	Oct. 07, 2020

The objectives of the first lab will be for you to set up a programming environment, specifically to have an integrated development environment (IDE) installed and functioning on your local drive in the lab or on your own desktop or laptop computer. Also, you will be developing on the program specification of the lab assignments that you will be gradually completing throughout this term.

Step 1. Environment Setup

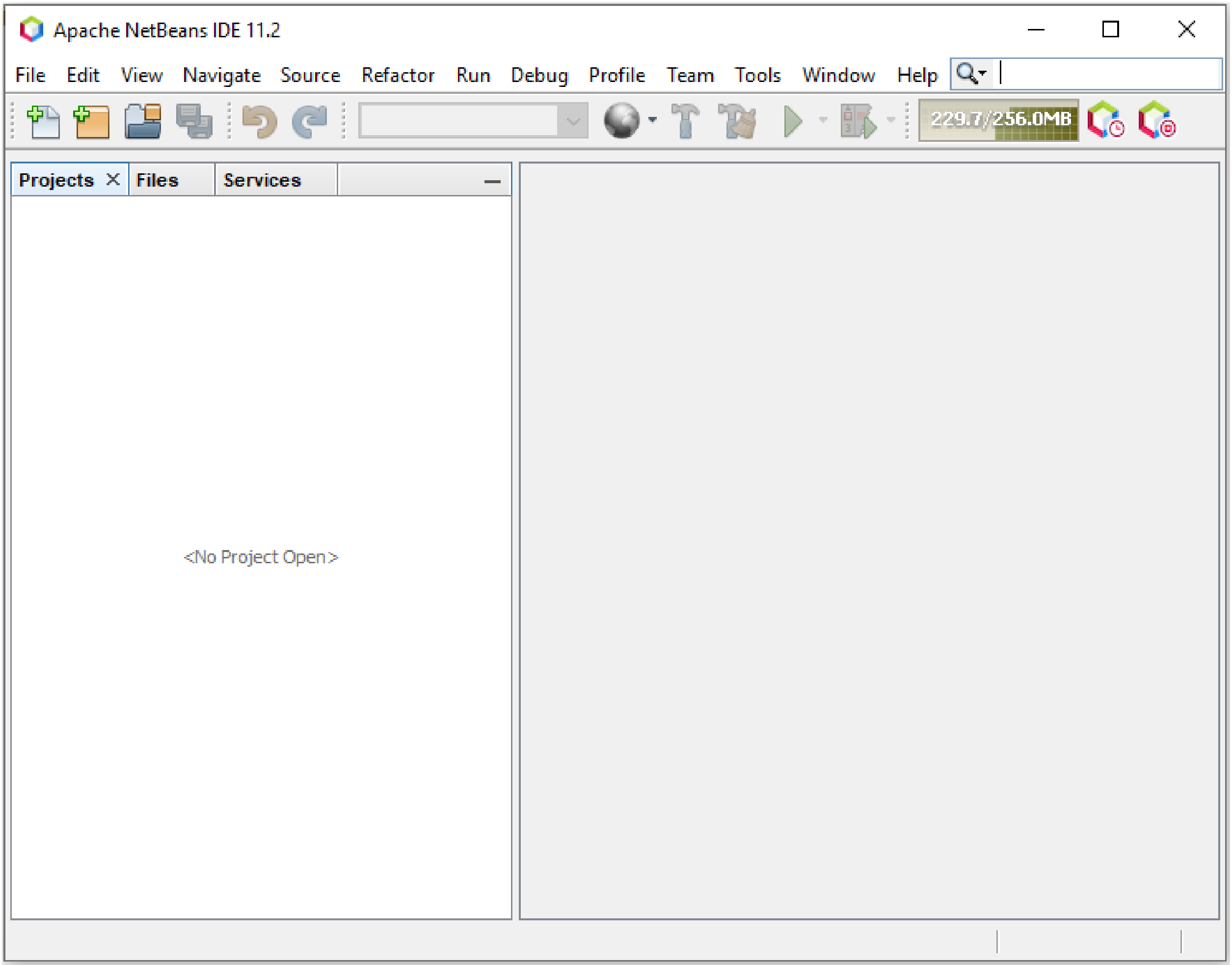
In computer engineering, there are hardware description languages (HDL) such as VHDL and Verilog that are specialized computer language used to describe the structure and behavior of electronic circuits, and most commonly, digital logic circuits. However, in computer science and this course mainly, we rely on schematic-based design methods that create a representation of functionality. We literally use pen and paper to draw circuits and gates. Therefore, we do not cover any HDL herein. We instead simulate the circuit's functionality via writing programs using a general-purpose programing language to make sure that our designed circuit is working correctly.

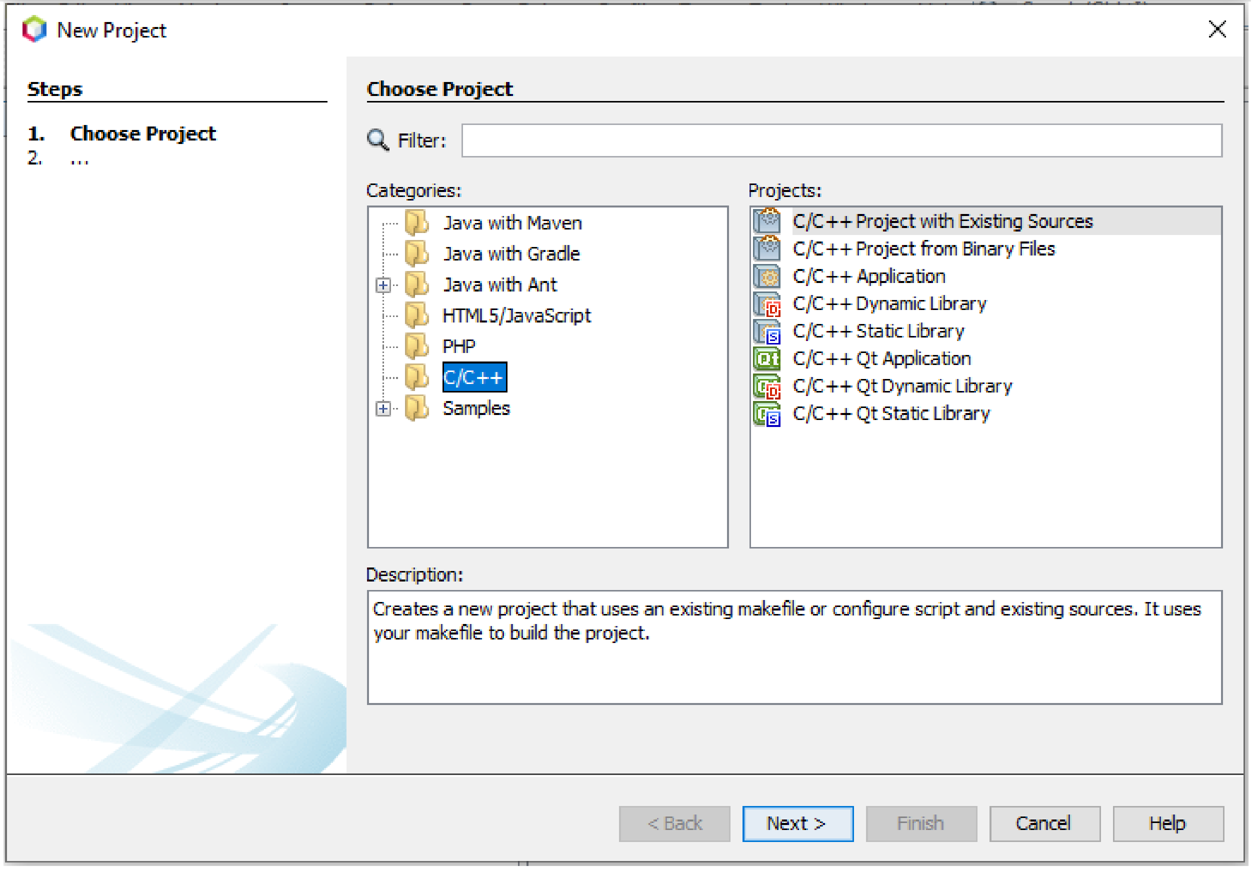
We will use NetBeans in this course as our IDE and C++ as the programming language. An alternative to IDE would be Eclipse, which is as powerful as NetBeans. Both are free and open-source.

Installing NetBeans or Eclipse is easy; go to the download page and grab the binary for your platform. For instance, for NetBeans:

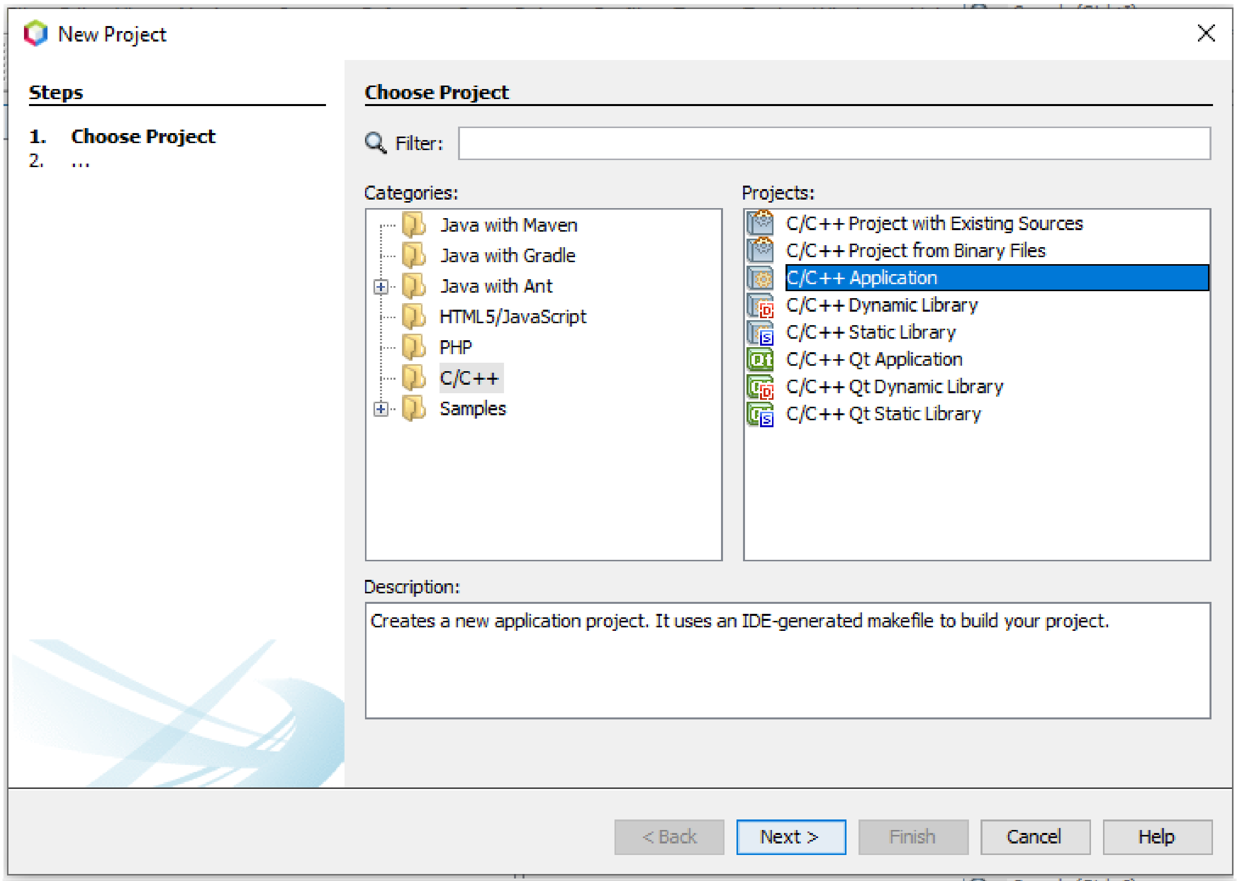
Windows: Apache-NetBeans-12.0-bin-windows-x64.exe
Linux: Apache-NetBeans-12.0-bin-linux-x64.sh
Mac: Apache-NetBeans-12.0-bin-macosx.dmg

When you've got it installed, either NetBeans or Eclipse, then make sure you can start up a command line and run it. Here's a quick test for you to try and create a new C++ project:


Apache NetBeans IDE 11.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
229.7/256.0MB
Projects
Files
Services
<No Project Open>



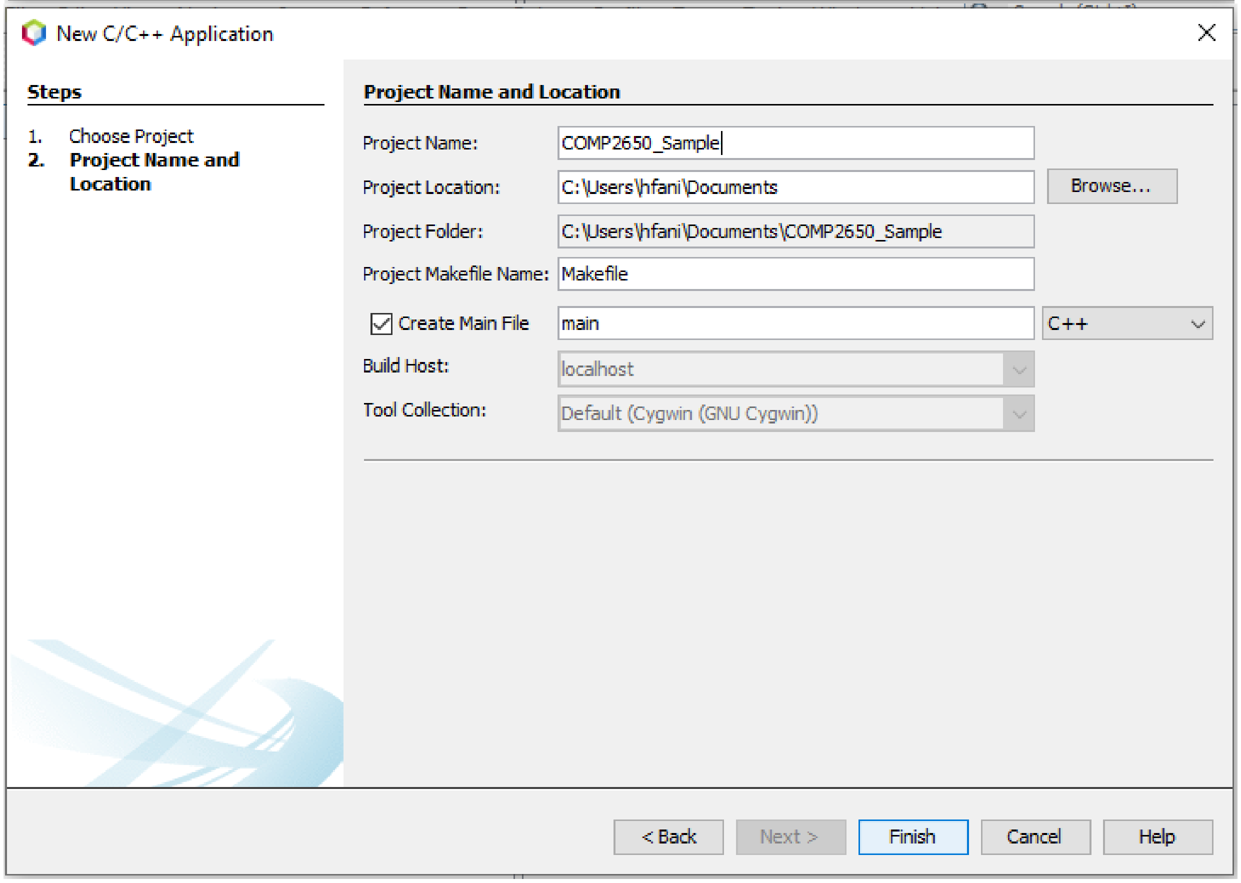
Creates a new project that uses an existing makefile or configure script and existing sources. It uses your makefile to build the project.



Categories: C/C++

Projects: C/C++ Application

Description: Creates a new application project. It uses an IDE-generated makefile to build your project.



Project Name: COMP2650_Sample

Project Location: C:\Users\hfani\Documents

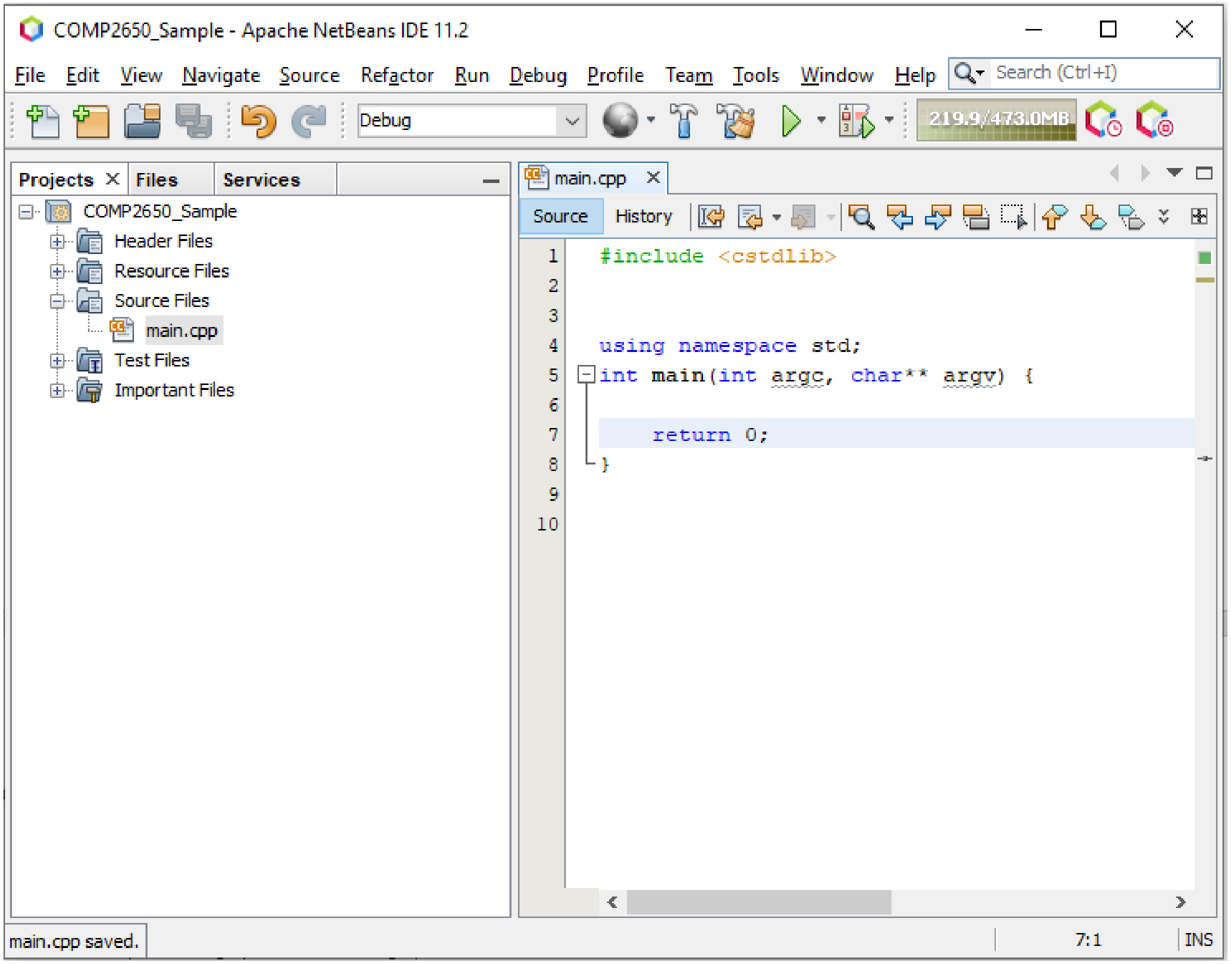
Project Folder: C:\Users\hfani\Documents\COMP2650_Sample

Project Makefile Name: Makefile

☑ Create Main File: main C++

Build Host: localhost

Tool Collection: Default (Cygwin (GNU Cygwin))



```
#include <cstdlib>


using namespace std;
int main(int argc, char** argv) {

    return 0;
}
```

main.cpp saved.

Step2. Writing the First Program

Now, let’s write a quick program that accepts two Boolean values and returns the AND of them. Recall from math, the truth table for AND operation, also shown as ‘∧’ or ‘&’ is as follows:

x	y	x AND y	x ∧ y	x & y
false	false	false	false	false
false	true	false	false	false
true	false	false	false	false
true	true	true	true	true

We know that C++ language has a type for boolean values, called `bool`, that `accept` false and `true`. However, for simplicity, let’s use the standard type int for our program and use 0 for `false` and 1 for `true` values.

```
01 #include <cstdlib>
02
03 using namespace std;
04 int main(int argc, char** argv) {
05
06     int x;
07     int y;
08     scanf("%d", &x);
09     scanf("%d", &y);
10     printf("%d AND %d is %d", x, y, x & y);
11     return 0;
12 }
```

Let’s build the project and run the program to see it’s working fine according to the above truth tables. Depending on what computer system and folder you created the program, you see the build messages similar to the followings in the `Output` tab → `(Build, Run)` subtab:

```
cd 'C:\Users\hfani\Documents\COMP2650_Sample'
C:\cygwin64\bin\make.exe -f Makefile CONF=Debug
"/usr/bin/make" -f nbproject/Makefile-Debug.mk QMAKE= SUBPROJECTS= .build-conf
make[1]: Entering directory '/cygdrive/c/Users/hfani/Documents/COMP2650_Sample'
"/usr/bin/make"  -f nbproject/Makefile-Debug.mk dist/Debug/Cygwin-
Windows/comp2650_sample.exe
make[2]: Entering directory '/cygdrive/c/Users/hfani/Documents/COMP2650_Sample'
mkdir -p build/Debug/Cygwin-Windows
rm -f "build/Debug/Cygwin-Windows/main.o.d"
g++    -c -g -MMD -MP -MF "build/Debug/Cygwin-Windows/main.o.d" -o build/Debug/Cygwin-
Windows/main.o main.cpp
main.cpp: In function 'int main(int, char**)':
main.cpp:8:19: error: 'scanf' was not declared in this scope
     scanf("%d", &x);
                   ^
main.cpp:10:42: error: 'printf' was not declared in this scope
     printf("%d AND %d is %d", x, y, x & y);
                                          ^
make[2]: *** [nbproject/Makefile-Debug.mk:68: build/Debug/Cygwin-Windows/main.o] Error 1
make[2]: Leaving directory '/cygdrive/c/Users/hfani/Documents/COMP2650_Sample'
make[1]: *** [nbproject/Makefile-Debug.mk:59: .build-conf] Error 2
make[1]: Leaving directory '/cygdrive/c/Users/hfani/Documents/COMP2650_Sample'
make: *** [nbproject/Makefile-impl.mk:40: .build-impl] Error 2

BUILD FAILED (exit value 2, total time: 1s)
```

As you can see, the build failed for our program due to 2 errors in finding `scanf` and `printf` functions. Because the linker could not find these functions declared in our project. Recall from the C++ program language course, and we know that these two functions are in the standard I/O library in `stdio.h`. Let's include this library and built the program again:

```
01 #include <cstdlib>
02 #include <stdio.h>

03 using namespace std;
04 int main(int argc, char** argv) {
05
06     int x;
07     int y;
08     scanf("%d", &x);
09     scanf("%d", &y);
10     printf("%d AND %d is %d", x, y, x & y);
11     return 0;
12 }
```

```
cd 'C:\Users\hfani\Documents\COMP2650_Sample'
C:\cygwin64\bin\make.exe -f Makefile CONF=Debug
"/usr/bin/make" -f nbproject/Makefile-Debug.mk QMAKE= SUBPROJECTS= .build-conf
make[1]: Entering directory '/cygdrive/c/Users/hfani/Documents/COMP2650_Sample'
"/usr/bin/make"  -f nbproject/Makefile-Debug.mk dist/Debug/Cygwin-
Windows/comp2650_sample.exe
make[2]: Entering directory '/cygdrive/c/Users/hfani/Documents/COMP2650_Sample'
mkdir -p build/Debug/Cygwin-Windows
rm -f "build/Debug/Cygwin-Windows/main.o.d"
g++    -c -g -MMD -MP -MF "build/Debug/Cygwin-Windows/main.o.d" -o build/Debug/Cygwin-
Windows/main.o main.cpp
mkdir -p dist/Debug/Cygwin-Windows
g++     -o dist/Debug/Cygwin-Windows/comp2650_sample build/Debug/Cygwin-Windows/main.o
make[2]: Leaving directory '/cygdrive/c/Users/hfani/Documents/COMP2650_Sample'
make[1]: Leaving directory '/cygdrive/c/Users/hfani/Documents/COMP2650_Sample'

BUILD SUCCESSFUL (total time: 2s)
```

The last green message shows that the project has been built successfully with no error. In the `(Run)` subtab, our program is running and waiting for the inputs, instructed by lines #08 and #09 of the program:

```
1
0
1 AND 0 is 0
RUN SUCCESSFUL (total time: 8m 17s)
```

Do run the program and try different inputs, e.g., 1 AND 1, 0 AND 0, 0 AND 1, and check whether the program correctly calculates the AND operation.

Lab Assignment

You should complete the above program under the name of a project COMP2650_Lab01_{UWinId} that firstly outputs a menu of four commands as follows:

```
Enter the command number:
   1) AND
   2) OR
   3) NOT
   4) Exit
```

Based on the chosen number of commands by the user, the program should then ask for the input(s). For instance, if a user selects (1), the program should accept two inputs as follows:

```
   x =
   y =
```

When the user enters the two boolean values (0 or 1), the program should apply the AND command on the input x and y and print the result and comes back to the main menu.

However, if the user selects (3), the program should ask for one input only:

```
   x =
```

When the user enters one boolean value, the program then applies the NOT operation on the input and prints out the result and comes back to the main menu. If the user selects (4), the program ends.

Deliverables

You will prepare and submit the program in one single zip file COMP2650_Lab01_{UWinID}.zip containing the following two items:

1. The entire project folder in NetBeans or Eclipse COMP2650_Lab01_{UWinID}
2. The result of the four commands in the file COMP2650_Lab01_Results_{UWinID}.jpg. Simply make a screenshot of the results and save it.
3. A lab report document in the PDF file COMP2650_Lab01_Report_{UWinID}.pdf. It should include:
   a. Your name, UWinID, and student number
   b. The description of the program that you attached, along with any prerequisites that are needed to build and run the program. *Please note that if your program cannot be built and run on our computer systems, you will lose marks.*

In sum, your final zip file for the submission includes 1 folder (entire project folder), 1 image (results snapshot) and 1 pdf (report). *Please follow the naming convention as you lose marks otherwise.* Instead of {UWinID}, use your own UWindsor account name, e.g., mine is hfani@uwindsor.ca, so,

COMP2650_Lab01_hfani.zip

- COMP2650_Lab01_hfani
- COMP2650_Lab01_Report_hfani.pdf
- COMP2650_Lab01_Results_hfani.jpg