**Topics for Today’s Lab**

* **Compiling C/C++ program using g++ and gcc**
* **Command Line Arguments**
* **Makefile**

**Compiling C/C++ program using g++ and gcc:**

**For C++:**

Command: g++ *source\_files*… -o *output\_file*

**For C:**

Command: gcc *source\_files… -o output\_file*

Source files need not be cpp or c files. They can be preprocessed files, assembly files, or object files.

The whole compilation file works in the following way:

Cpp/c file(s)🡪Preprocessed file(s) 🡪 Assembly File(s) Generation 🡪 Object file(s) Generation 🡪 Final Executable

Every c/cpp file has its own preprocessed file, assembly file, and object file.

1. For running only the preprocessor, we use

-E option.

2. For running the compilation process till assembly file generation, we use –S option.

3. For running the compilation process till object file creation, we use –c option.

4. If no option is specified, the whole compilation process till the generation of executable will run.

A file generated using any option can be used to create the final executable. For example, let’s suppose that we have two source files: math.cpp and main.cpp, and we create object files:

g++ main.cpp –c –o main.o

g++ math.cpp –c –o math.o

The object files created using above two commands can be used to generate the final executable.

g++ main.o math.o –o my\_executable

The file named “my\_executable” is the final exe file. There is specific extension for executable files in Linux.

**Command Line Arguments:**

Command line arguments are a way to pass data to the program. Command line arguments are passed to the main function. Suppose we want to pass two integer numbers to main function of an executable program called a.out. On the terminal write the following line:

./a.out 1 22

./a.out is the usual method of running an executable via the terminal. Here 1 and 22 are the numbers that we have passed as command line argument to the program. These arguments are passed to the main function. In order for the main function to be able to accept the arguments, we have to change the signature of main function as follows:

int main(int argc, char \*arg[]);

* argc is the counter. It tells how many arguments have been passed.
* arg is the character pointer to our arguments.

argc in this case will not be equal to 2, but it will be equal to 3. This is because the name ./a.out is also passed as command line argument. At index 0 of arg, we have ./a.out; at index 1, we have 1; and at index 2, we have 22. Here 1 and 22 are in the form of character string, we have to convert them to integers by using a function atoi. Suppose we want to add the passed numbers and print the sum on the screen:

cout<< atoi(arg[1]) + atoi(arg[2]);

**Question 1:** Write a C or C++ program that accepts a file name as command line argument and prints the file’s contents on console. If the file does not exist, print some error on the screen.

**Question 2:** Write a C or C++ program that accepts a list of integers as command line arguments sorts the integers and print the sorted integers on the screen.

**Question 3:** Generate Makefile for the given source files. main.cpp and utility.cpp. Generate object files first, and then generate the final executable.