Creative Software Design

1 - Lab2 - g++, make, gdb

Yoonsang Lee
Fall 2023

Introduction

- Lab2 TA: Eunho Jung (정은호)
  - jho6394@hanyang.ac.kr

- Lab2 Undergraduate Mentor: Minwoo Park (박민우)

Outline

- G++
- Make
- GDB
- Time for Assignment 1-1

What is G++ ?

- Open-sourced C++ compiler

- Most formats and options are the same as the default C compiler (cc)
  - **g++** [options] <infile> ...
    - -c : compile and assemble, but do not link Create only object file (.o) without creating executable
    - -g : debug info. Contains information necessary for debugging (source code, etc.)
    - -o <outfile> : Place the output into <outfile>
    - -I<dir> : include directory. (directory name to look for headers when compiling)
    - -L<dir> : library directory. (Directory name to look for library files when linking)
    - -D<symbol>[=def] : define a macro to use at compile time
    - ... : There are numerous other options.

Example : Compile & Link

- Write main.cpp, print.cpp, print.h

```
// main.cpp
#include "print.h"
int main() {
  print_hello();
  return 0;
}
```

```
// print.cpp
#include <iostream>
void print_hello() {
  std::cout << “hello world!” << std::endl;
}
```

```
// print.h
void print_hello();
```

Example : Compile & Link

- Compile and link the two source files (main.cpp, print.cpp)

(Shell – working directory)

```
$ g++ -c -o main.o main.cpp
$ g++ -c -o print.o print.cpp
$ g++ -o hello_world main.o print.o
```

(Shell – working directory)

```
$ g++ -o hello_world main.cpp print.cpp
```

- Run the created executable

(Shell – working directory)

```
$ ./hello_world
```

Compile & Link

- This is a very brief introduction on how to compile and link using g++.

- The details of compile & link process will be covered in the lecture "5-Compilation and Linkage, CMD Args".

Make

- Build tools that have been around for a long time on Unix operating systems
  - Rules for how to compile and link the source to create an executable

Makefile

- When “make” is run, find Makefile (or makefile) in that directory and runs it as usual

- How to write Makefile

```
target: prerequisites
<TAB>command1
<TAB>command2
```

  - target : File or state to create(such as.o or excutable) 등)
  - prerequisites : List of files needed to create target
  - command(s) :Each step command to create a target. <Tab> must be placed before the command.

Example: Writing / Running makefile

- Write makefile

(Shell – working directory)

$ vi Makefile

```
hello_world: main.o print.o
	g++ -o hello_world main.o print.o
main.o: main.cpp
	g++ -c main.cpp
print.o: print.cpp
	g++ -c print.cpp
clean:
	rm hello_world main.o print.o
```

Example: Writing / Running makefile

- Execute makefile (1) : generate executable file

```
(Shell – working directory)
$ make
```

- Execute makefile (2) : Remove Excutable file and All object files

```
(Shell – working directory)
$ make clean
```

GDB

Debugging tools - help you find the wrong parts of your program by checking its status when the program is running or when it crashes.

When you build a program, you need to give it the -g option to see the information you need.

gdb [options] <command>

- <command> : If the current directory is not in your PATH, you must include ./.
- Basic command
  - r [arguments] : Run the given command.
  - bt : backtrack. Show current call stack status.
  - up/down [steps] : Move up / down a given step from the current position of the call stack.
  - p <variable> : Display the value of a given variable.
  - q : exit gdb process.
  - Use more easy-to-use improved programs such as cgdb and ddd

Example

(Shell – working directory)

$ vi test.cc

```
void IncorrectAccess(int* array, int i, int n) {
  if (i < n) {
    array[i] = 0;
    IncorrectAccess(array, i + 1, n);
  }
}
int main() {
  int array[10];
  IncorrectAccess(array, 0, 20);
  return 0;
}
```

(Shell – working directory)

$ g++ -o test test.cc
$ gdb ./test
…
(gdb)
…

Assignment 1-1

- Now, let's start the assignment 1-1.
- Assignment 1-1 is just for practice, will not be included in the final grade.

- However, you need to complete and submit your answers to figure out **how to set up the environment** and **how to submit your assignments**.

- LMS course home – Lecture Contents (강의콘텐츠) – 1st week (1주차) - Assignment1-1

Assignments

- TAs and undergraduate mentors will help you to solve the problems.
  - You can ask questions!
  - Sitting next to you and **debugging together is not the role of TAs and mentors**. They can explain what to look for, but they don't debug together.
  - TAs and mentors are not someone who catches fish for you; **they are someone who teaches you how to catch fish.**

- Lab session policy
  - You can leave the room after completing all the assignments and getting confirmation from the TA.
  - Even if you haven't completed all the assignments, you can leave after 1.5 hours from the start of the lab session.