Creative Software Design

1 - Course Intro

Yoonsang Lee
Fall 2023

Course Information

- Instructor: Yoonsang Lee (이윤상)
  - yoonsanglee@hanyang.ac.kr

- TA:
  - Lab1: Bokyeong Jang (장보경), alswl7763@hanyang.ac.kr
  - Lab2: Eunho Jung (정은호), jho6394@hanyang.ac.kr

- Course Hompage
  - The LMS course homepage at portal.hanyang.ac.kr (or learning.hanyang.ac.kr)
  - Slides will be uploaded to Lecture Contents (**강의콘텐츠**)**,** probably *just before the lecture*. So, **download lecture slides at the beginning of each lecture.**
  - If you want to study the lecture slides **in advance**, please refer to last year's lecture slides (They won't change much): https://cgrhyu.github.io/courses/2022-fall-csd.html

Course Overview

- In this course, you will
  - Learn the fundamentals of C++ language
    - key concepts of object-oriented programming such as classes, inheritance, and polymorphism
    - references, pointers, dynamic allocation
  - Practice programming skills by writing many exercise programs
  - Practice using development tools and editors in Unix/Linux environment.

References

- Beginner's book:
  - C++ Primer Plus (6th edition), Sthephen Prata
  - But this is **not the textbook** (No official textbook for this course).

- For deeper understanding:
  - Effective C++, Scott Meyers
  - More Effective C++, Scott Meyers
  - Effective STL, Scott Meyers
  - Effective Modern C++, Scott Meyers

Prerequisites

- Introduction to Software Design (소프트웨어 입문 설계)

- C programming language

- However, we'll have a brief review of C pointers / structures next week.

Schedule (subject to change)

Week	Topic	Tue	Wed	Thu
1	1 - Course Intro / 1 - Lab1 - Environment Setting, Vim 1 - Lab2 - G++, Make, GDB	9/5	9/6	9/7
2	2 - Review of C Pointer, Const and Structure	9/12	9/13	9/14
3	3 - Differences Between C and C++	9/19	9/20	9/21
4	4 - Dynamic Memory Allocation, References	9/26	9/27	9/28
5	5 - Compilation and Linkage, CMD Args	10/3	10/4	10/5
6	6 - Class	10/10	10/11	10/12
7	7 - Standard Template Library (STL)	10/17	10/18	10/19
8	**Midterm Exam**	10/24	10/25	10/26
9	8 - Inheritance, Const & Class	10/31	11/1	11/2
10	9 - Polymorphism 1	11/7	11/8	11/9
11	10 - Polymorphism 2	11/14	11/15	11/16
12	11 - Copy Constructor, Operator Overloading	11/21	11/22	11/23
13	12 - Template	11/28	11/29	11/30
14	13 - Exception Handling	12/5	12/6	12/7
15	**Final Exam**	12/12	12/13	12/14

Lectures & Labs

- Lecture (Tue) + Labs (Wed, Thu)

- Lecture (by instructor)
  - Traditional classroom-based learning.

- Labs (by TAs)
  - Time for solving assignment problems by yourselves.
  - TAs and undergraduate mentors will help you.

- Lectures and labs will be given in-person.

Assignments

- One assignment for each lab session.

- Lab1(Wed) assignment due: 23:59 on the day.

- Lab2(Thu) assignment due: 23:59 on next Mon.

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.

Policy for Assignments

- **NO SCORE** for late submissions
  - Submit before the deadline!

- **NO SCORE** for copying
  - If A copies B's code, A and B will get 0 point.
  - If A, B, C copies the same code from the internet, they will all get 0 point.
  - Collaboration is encouraged, **but assignments must be your own work.**

Grading

Midterm exam	40%
Final exam	40%
Assignments	15%
Quiz	3%
Attendance	2%

- You will get **"F"** for more than **5 absences in lectures or 10 absences in labs**.
  - 3 lates are regarded as 1 absence.

- **Absences from the midterm or final exam → F**

- CAUTION: Grade penalty:
  - **4th year** (senior) students: **max grade A0**

Grading Policy

- Basic principle: Separating the grades where there is a big gap between points.

- Guideline:

A	20%~25%
B	25%~35%
C	40%

Language

- This is **not** an English-only lecture. I will use Korean.

- But all lecture / lab slides and assignments are written in English.

- Midterm / final exams are written in English as well.

Computer Science

YS what is computer science?

Computer science is the study of computers, computational systems, and the algorithms and processes used to manipulate and store data. It encompasses both the theoretical and practical aspects of computing, ranging from the design and analysis of algorithms to the development of software and hardware systems.

Computer science deals with various topics, including programming, data structures, artificial intelligence, computer graphics, database management, cybersecurity, and more. It plays a fundamental role in modern technology and has applications in nearly every aspect of our daily lives, from smartphones and web applications to scientific research and business operations.

In essence, computer science is a discipline that explores how computers work, how they can be programmed to perform specific tasks, and how they can be used to solve complex problems across different domains.
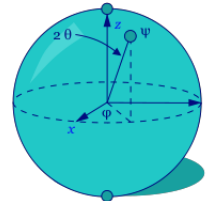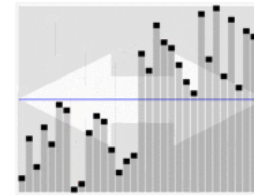
Areas of Computer Science
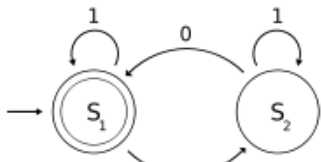
Automata theory
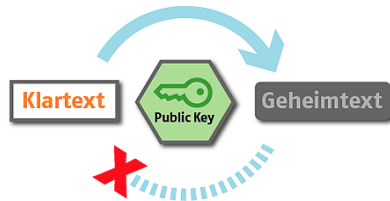
Cryptography

Quantum computing

Algorithms

Data structure

Programming language

Compiler design

Operating system
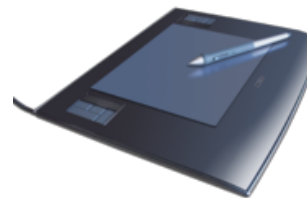

Database

Computer network
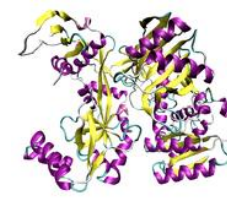

Machine learning

Computer vision

Robotics

HCI

Bioinformatics

Computer graphics

Programming

Programming is the comprehensive process that leads from an original formulation of a computing problem to executable programs. [wikipedia]

Let's revisit the basics related to programming.

- Programming language.

- Computer hardware.

- Operating system and development environment.

- Data structure and algorithm.

- Coding style, collaboration, source management (version control).

Programming Language

A formal language designed to communicate instructions to a computer.

- Syntax and semantics.

    - Language syntax and constructs.

    - Type checking - strongly typed languages.

- Standard library and running environment.

    - Tightly related to the operating system and compiler.

In this class, we learn/use ...

- C++ programming language

- Linux operating system

- g++ (compiler), make & CMake (build tools), gdb (debugger), and more

Computer Hardware

- Von Neumann architecture.

  ○ Main components are CPU and memory.

  ○ Both program and data are stored in the memory (stored-program computer).

  ○ When a program is executed, the instructions are fetched from the memory, then decoded for execution.

  ○ Those instructions consist of operations that modify the values in specific memory locations.
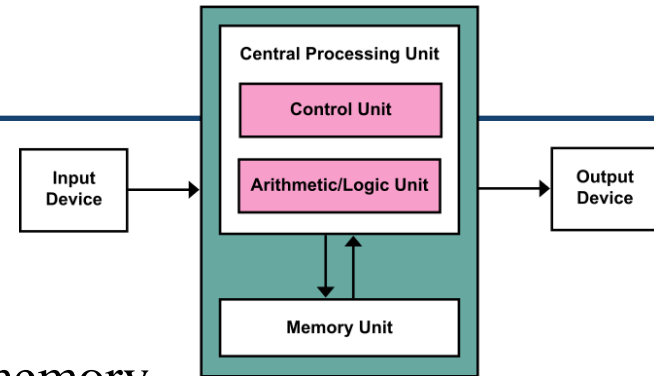
- Virtual memory.

  ○ The memory space that a program sees is a continuous linear address space.
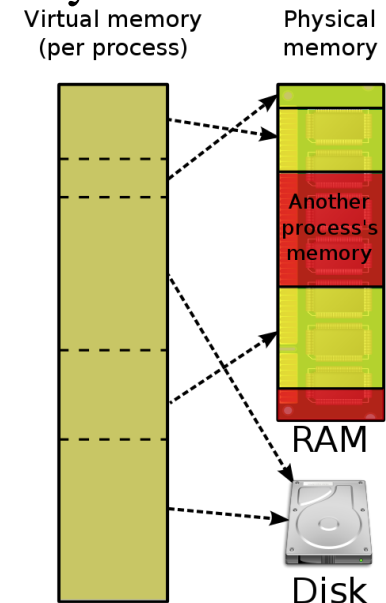
Operating System and Dev. Environment

- The operating system is in charge of

  - resource management, including CPU and memory (time-sharing and virtual memory),

  - input/output (I/O) operations and file management, and

  - execution and termination of programs.

- Development environment

  - Text editor, basic text tools +
    Compiler, linker, debugger, profiler +
    Build system, source management tools, etc.

  - vim, emacs, VS Code, grep, find, diff, …
    g++, gcc, gdb, cgdb, …
    make, CMake, git, svn, ...

  - IDE (Integrated -) : Visual Studio, Eclipse, IntelliJ, ...

Development Environment in This Class

- Operating System: Ubuntu

- Compiler & Linker: g++

- Build system: make, CMake

- Assignments will be graded in this environment.
  - Even if it builds and runs without any problem in another OS (ex. Windows), **you will not receive a score if it does not build and run in this "standard" environment.**

Data Structure and Algorithm

Programming starts with designing data structures and algorithms to solve the given problem.

- Data structure = how is the information stored?

  - Array (fixed-size, dynamically allocated)

  - Linked list (doubly-), stack, queue, deque,

  - Strings,

  - Trees (binary, B-), graphs, ...

- Algorithms = how is the information processed?

  - Sorting, searching, matching (regular expression),

  - Keeping a tree balanced, path finding in a graph, etc.

Collaboration and Source Management

Large programs are almost always built by multiple programmers over long period of time.

- Coding style = how to format the code?

  - Nothing to do with the program performance, but

  - Very important for human programmers to easily collaborate.

  - Tabs vs spaces, 80-column or not, blanks before/after operators, braces in a new line or in the same line, etc.

- Source management (or version control) system - git, svn, ...

  - Share the code with multiple programmers and manage multiple versions.

  - Review the changes and merge them without breaking the existing system.

  - Release management, etc.

Interactive C++

- Interactive environment helps learning a programming language, libraries, and other tools.

- Cling

  - https://github.com/root-project/cling

  - Available on Linux, Mac, and (as a beta release) on Windows.

- C++ Tutor

  - http://www.pythontutor.com/cpp.html#mode=edit

  - Visualizes the execution of C++ program

Interactive C++

- C++ Tutor Demo

```cpp
#include <iostream>

int main() {
  std::cout << "hello_world\n";  // Print hello_world.
  int a = 1;
  int b = 2;
  int c = a + b;
  return 0;
}
```

Questions

- After lecture, if you have questions, ask on the "Q&A Board" ("문의게시판") of the LMS course home.

  – TA will check and respond at least once a day.

- In lecture, we'll use an online, anonymous Q&A platform - slido.com - to encourage questions.

Just Try Asking a Question!

- Go to https://www.slido.com/

- Join **#csd-ys**

- **Do not bookmark a slido event page** because new events will be created every week!

- Ask any questions**!**

Questions – Slido.com

- We'll use the slido Q&A **only during lecture time.**
  - Not after lecture time
  - Not in lab sessions
  - No written answers

- Please ask questions **anonymously**.
  - Just leave your name blank when post a question.

Quiz – Slido.com

- Three quiz problems per each lecture (using slido.com Polls).

- Very simple questions – you have to submit the answer in two minutes.

Quiz – Slido.com

- Your quiz score is based solely on whether or not your answers is submitted in the following format:
  - **Student ID: Your answer**
  - **e.g. 2023123456: 4)**

- **Whether your answer is correct or not has nothing to do with your quiz score!**

- This year, attendance check has nothing to do with quiz submissions.

Just Try a Quiz!

- Go to https://www.slido.com/

- Join **#csd-ys**

- Click "Polls"

- Submit your answer in the following format:
  - **Student ID: Your answer**
  - **e.g. 2023123456: 4)**

- Note that your quiz answer must be submitted **in the above format** to receive a quiz score!

About Laptop

- Lecture
  - The lecture slides contains many C++ code.
  - During lectures, you can compile, run, and test the code on your laptop.

- Lab
  - In labs, you will need to bring your own laptop for practice and assignments.
    - If you would like to rent a laptop, please contact the TA by email by the next week lab. However, it is recommended that you bring your own laptop, as they will be quite old.

Classroom Etiquette

- **DO NOT negatively affect other students** in the classroom. For example,
  - Doing other things (e.g. games) with your computer
  - Using your phone for a long time
  - Private conversation
  - Sleeping on a desk

Lastly...

- If you agree on all these policies, see you this week's lab session!

- If not, you can consider taking other classes instead.

[Lab] Downloads in advance

- Before joining the first lab session (tomorrow), download the following files in advance.

- VirtualBox: http://www.virtualbox.org

- Ubuntu Desktop image (in your USB memory stick): http://releases.ubuntu.com/22.04

- You don't need to download them if you're already using Ubuntu on your laptop.

Next Time

- Labs in this week:
  - 1 - Lab1 - Environment Setting, Vim
  - 1 - Lab2 - G++, Make, GDB

- Next lecture:
  - 2 - Review of C Pointer, Const and Structure