Introduction

September 4, 2019
1. Course Introduction
2. Programming Overview
3. History of C Language
4. Editor
5. Intro of Linux Command-line
The course lecture materials I am using are mostly based on the professor Ming Ouyang’s past CS240 course materials. I am much grateful for his gracious help and guidance.

However, I made some changes based on my preferences.
The final grade is determined by 40% of homework and 60% of tests. There will be three in class paper tests. For how many homework, we will see how class progress during semester. We will see if I will drop the lowest homework grade for all based how class progress. If you understood course topics and done the homework well, you will get good grades.
Announcements and discussions will be posted on Blackboard
Check Blackboard and read your email regularly
Plagiarism

- Don’t do it in any form or it will be reported.
- Test cheating will lead to fail.
Done?
Access Linux server.

Transfer files between your machine and sever.
For the submission of homework, I will note it in each homework.

"K&R" is often being referenced to either this book or these two computer scientists, cause they two are just that important and great.

"K&R" created the C language in order to write UNIX system efficiently.

You could just search the bio of them to learn a lot about the early history of programming language.
Reading Suggestions

- Read the textbook ahead of the class, but don’t expect to understand everything.
- Skip preface. Skip or skim through introduction.
- You will understand better introduction after you are familiar with lots of the concepts of C.
Now, let’s getting into some basic computer principle.

- CPU executes on lines of binary digits or bits, called machine language or machine code or object code.
- Machine language is the only language a computer is capable of understanding.
- To see a binary code, you could type in linux terminal `xxd file-name` for hexadecimal encoding or `xxd -b file-name` for binary encoding.
- If you open a compiled C or Java program in a text editor like sublime text, you will see the lines of hexadecimal bits, not English characters, because a binary file contains the information about its way of encoding.
Programming Overview

- How a line of binary digits is executed is determined by a CPU’s instruction set architecture (ISA) which is also referred to as architecture or computer architecture.
- Let’s take an ‘example’ for a figurative sense of understanding.

```
00000001 00000002 00000003
```

Let’s set
- 00000001 means addition for a ’dummy’ CPU
- 00000002 and 00000003 means address.

Then this ’dummy’ CPU executes this line of binary digits, it adds the data in addresses of 00000002 and 00000003 means.

Again this example is for a intuitive understanding of how CPU runs binary codes, it is by no means right in any sense.
Take a look at Punched Tape Codes in early days, which means binary code on tape. http://www.chilton-computing.org.uk/acl/literature/chapman/p015.htm

However, it is impractical for programmers to write programs directly in binary digits.

In reality, programmers write program in English character with certain rules tied to ISA, which are stipulated by different compilers, i.e. different programming languages, like C, Java, etc..

And the text code written by programmers is called source code.

So as a programmer, you will work on writing source code and then compile it for executing.
The main processes of a modern compiler are working like these:

source code $\rightarrow$ preprocessor $\rightarrow$ compiler $\rightarrow$ assembler $\rightarrow$ object code $\rightarrow$ linker $\rightarrow$ executables

In reality, the whole processes are divided into source code writing and compiling.

Let’s take two examples of ”Hello world!” in C and Java to see what this means.
Compiler

- gcc -o hello hello.c
- ./hello
- javac hello.java
- java hello

- ‘.’ means current folder; ‘./hello’ means executing binary file hello in current folder directly on OS.
- ‘java hello’ means executing Java binary file in current folder within Java Virtual machine.
Some language called script language, like bash, python, integrates compiling and executing processes together so that source code can be directly executed.

For example, python hello.py
Let’s talk about a little bit of the history of C.

- The C programming language was devised in the early 1970s by Dennis M. Ritchie an employee from Bell Labs (ATT).
- It was actually created to move the UNIX kernel code from assembly to a higher level language, which would do the same tasks with fewer lines of code.
- The UNIX operating system which started in 1969, then was rewritten in C in 1972.
- Since it’s birth, C is closely related to operating system.
Linux is directive derivative of Unix, which is written mostly in C, with some parts in assembly.

Linux was intended for personal computers based on the Intel x86 architecture and free of using. Now about 97 percent of the world's 500 most powerful supercomputers run the Linux kernel.

Microsoft's Windows kernel and Mac OS X kernel are also developed mostly in C, with some parts in assembly language.

Every program and driver in a Mac, as in Windows and Linux computers, is running on a C-powered kernel.

Besides the world's most popular databases, including Oracle Database, MySQL, MS SQL Server, and PostgreSQL, are coded in C and C++.
Since C programs can be run directly on OS system, they are as fast as programs can be.

A speed comparison.
https://leetcode.com/problems/two-sum/submissions/

For all implementations of the same algorithm in different languages, the C implementation is the fastest.

It is mostly because other languages have to run in a Virtual Machine resides on operating system other than on os itself.
Editors on Linux

- **vi**
  - Installed by default on typical Linux installations
  - Popular among system admins
  - vim is a variant
  - A tutorial
    https://www.tutorialspoint.com/unix/unix-vi-editor.htm

- **emacs**
  - Installed by this command:
    `sudo yum install emacs`
  - A lot bigger and a little slower than vi
  - A guided tour http://www.gnu.org/software/emacs/tour/

- On our department server, there is also a simpler editor Nano.
- Vi/vim and emacs are professional and come well along with linux system.
- However they are time consuming to learn and frustrating to use for beginners.
Let’s do a simple tutorial.
Nano: this is simple.
Basic commands

Let’s do a simple tutorial.
https://www.git-tower.com/blog/command-line-cheat-sheet
Begin to read the first chapter, and try the examples on your home machine or server!