Lecture 2

- Linux/shell/emacs/g++ Intro
- How g++ roughly works
- Customizing shell and emacs
- Basic C++ building blocks

UNIX Filesystem

- Hierarchical **(tree structure)**
- / represents the root directory
- Directories ("folders") can contain other directories and files **(internal nodes)**
- Files **(leaves)** are just sequences of bytes
- Files/directories are uniquely located by a directory path. E.g. /home/user/AS1/foo.c
  - / is also used as directory separator

Command Shell

- Command line interface (text window with keyboard attached to it) e.g. "xterm"
- Issue operating system commands directly via keyboard input; e.g.
  - **ls** (list directory contents)
  - **cd** (change directory) "cd workdir"
  - **mv** (move/rename) "mv old-file new-file"
  - **mkdir** (create directory) "mkdir AS1"
  - **cp** (copy file or directory) "cp -r dir backup"
  - **rm** (remove file or directory) "rm -rf dir"
  - **man** (command info) "man man"

Shell Basics

- special directories:
  - / root directory everything is stored here
  - . current directory "cp ./foo ./bar" = "cp foo bar"
  - .. parent directory
  - ~ home directory "cd ~/foo"
- command history/editing
  - use arrow keys to navigate, delete key
- simple programming language
  - variables, functions, command aliases
- startup code in ~/.bashrc (when bash is used)
  - customizations! function ll() { ls -l "$@"; }

Getting Started

- In command shell issue: echo $SHELL
  - prints contents of variable $SHELL to screen
  - possible outputs: /bin/bash /bin/tcsh /bin/csh ...
- UNIX commands are stored in /bin and /usr/bin
- To learn more about your default shell
  - “man bash” or “man tcsh” ...
- Take your time to **RTFM**! It pays off!
- Customize ~/.bashrc (or ~/.tcshrc ...)
  [sample file on my webpage]

Launching Programs

- type program name (+ parameters) and hit the return key. E.g.
  - ls -l <return-key>
  - emacs foo.c <return-key>
- Shell interprets the first word as command name and tries to locate a function definition with this name (see ~/.bashrc). If this fails it searches in the directories listed in variable $PATH (try echo $PATH)
- detach program from terminal (running in background): command & (= ctrl-z + bg)

Edit Textfiles

- Many editors exist: emacs, vi, vim, ...
- emacs is the most powerful
- Type “emacs x <return-key>” to edit file x
- Huge number of commands bound to keys
  - ctrl-x ctrl-s save buffer; ctrl-x ctrl-f load file
  - ctrl-x ctrl-c exit; ctrl-s search
  - alt-% search and replace; ctrl-x 2 split window
  - alt-x command launch external commands such as gdb, gnus
- “man emacs”, emacs reference cards, emacs tutorial (in help menu or on the web)
- Customization: edit ~/.emacs [sample]

Customizing the shell and emacs

- edit file **~/.bashrc** (bash) or **~/.tcshrc** (tcsh)
- to customize emacs edit **~/.emacs**
- sample files are on the section web page
- no need to know everything ... as a start, just look at the comments to find the parts you want to adjust

First Program: hello.c

- Create file hello.c using emacs and save it

```
#include <iostream>
using namespace std;

int main()
{
  cout << “hello world” << endl;
  return 0;
}
```

- g++ -o hello hello.c generates executable hello which prints “hello world” after being invoked by issuing ./hello
- without -o option, g++ creates executable a.out