* Notes adapted by Doug Bowman from previous work by other members of the CS faculty at Virginia Tech
What is Unix?

- A modern computer operating system

- Operating System
  - “a program that acts as an intermediary between a user of the computer and the computer hardware”
  - Software that manages your computer’s resources (files, programs, disks, network)
  - Examples: Windows, MacOS, Solaris, BSD, Linux (e.g. Mandrake, Red Hat, Slackware)

- Modern
  - Stable, flexible, configurable, allows multiple users and programs
Why Unix?

- Used in many scientific and industrial settings
- Huge number of **free** and **well-written** software programs
- Excellent programming environment
- Internet servers and services run on Unix
  - Roughly 65% of the world’s web servers are Linux/Unix machines running Apache
Ken Thompson and Dennis Ritchie originally developed the earliest versions of Unix at Bell Labs for internal use in the 1970s

- Simple and elegant
- Meant for programmers and experts
- Written in a high-level language instead of assembly language
  - Small portion written in assembly language (kernel)
  - Remaining code written in C on top of the kernel

http://www.bell-labs.com/history/unix/
Unix Variants

- Two main threads of development
  - Unix System Laboratories ([http://www.unix.org](http://www.unix.org))
- Sun: SunOS, Solaris
- SGI: Irix
- FreeBSD, OpenBSD, NetBSD
- Hewlett-Packard: HP-UX
- Apple: OSX (Darwin)
- Linux (many flavors)
Brief History of Linux

- Andrew Tanenbaum, a Dutch professor developed MINIX to teach the inner workings of operating systems to his students
- In 1991 at the University of Helsinki, Linus Torvalds, inspired by Richard Stallman’s GNU free software project and the knowledge presented in Tanenbaum’s operating system, created Linux, an open-source, Unix-like operating system
- Over the last decade, the effort of thousands of open-source developers has resulted in the establishment of Linux as a stable, functional operating system
- [http://www.linuxgazette.com/node/9721](http://www.linuxgazette.com/node/9721)
Layers in a Unix-based System

- **Hardware**
  - CPU, memory, disks, terminals, etc.

- **Unix Operating System**
  - (process/memory management, file system, I/O)

- **Standard Library**
  - (open, close, read, write, etc.)

- **Standard Utility Programs**
  - (shells, editors, compilers, etc.)

- **System calls**

- **Library Interface**

- **User Interface**

- **Users**

user mode

kernel
Unix Structure

- The **kernel** is the core of the Unix operating system, controlling the system hardware and performing various low-level functions. Other parts of a Unix system (including user programs) call on the kernel to perform services for them.
- The **shell** accepts user commands and is responsible for seeing that they are carried out.
- The **filesystem** organizes all of the information on the computer and provides access to it for programs.
Over two hundred *utility* programs or *tools* are supplied with the Unix system. These utilities (or commands) support a variety of tasks such as copying files, editing text, performing calculations, and *developing software*.

This course will introduce a limited number of these utilities and tools, focusing on those that aid in software development.
Getting Started

- Logging in to a Unix machine requires an account on that system
- After logging in, some information about the system will be displayed, followed by a shell prompt, where commands may be entered
  - $
  - %
  - #
  - username@hostname>
  - hostname%

- HCI note: “tyranny of the blank screen”
The **shell** is the program you use to send commands to the Unix system

- Some commands are a single word
  - `who`
  - `date`
  - `ls`

- Others use additional information
  - `cat textfile`
  - `ls -l`
Command Syntax

- Commands must be entered exactly
- Be careful! Some commands can be very destructive. (e.g. `rm junk*` vs. `rm junk *`)
- Syntax: **command options argument(s)**
- **Options** modify a command’s execution
- **Arguments** indicate on what a command should act (often filenames)
Example Command: `ls`

- `ls -l`
- `ls -a`
- `ls -la`
- `ls -a; ls -l`
- `ls textfile1`
- `ls directory`
- `ls textfile1 textfile2`
- `ls -al textfile1`
If you don’t see a shell prompt...

- A program is probably running
- If you see a special program prompt, try to quit the program (*quit*, *bye*, *exit*)
- If you see nothing, you can usually
  - Stop the program with CTRL-z (program will wait until started again)
  - Interrupt the program with CTRL-c (program will usually die)
Ending your session

- **Always** log out when you are done
- Use the `exit` command to log out of a shell
- **Note**: If you are running in a windowing environment, logging out of a shell only ends that shell. You must also log out of the windowing system, typically selecting an option from a menu.