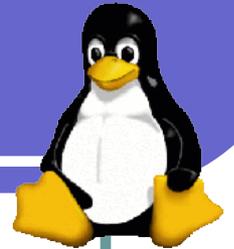


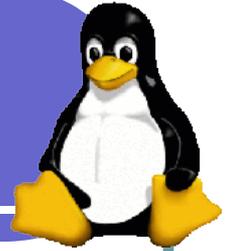
Introduction to Unix



Class 1

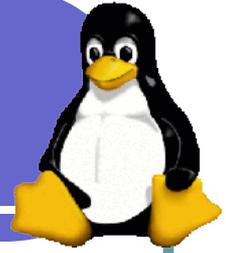
* Notes adapted by Alexey Onufriev from previous work by other members of the CS faculty at Virginia Tech

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, MacOSX, Solaris, BSD, Linux (e.g. Mandrake, Red Hat, Slackware, SUSE)
- Modern
 - Stable, flexible, configurable, allows multiple users and programs

Why Learn Unix?

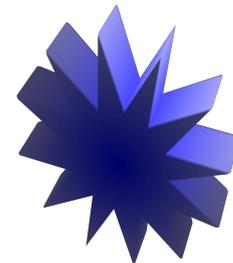
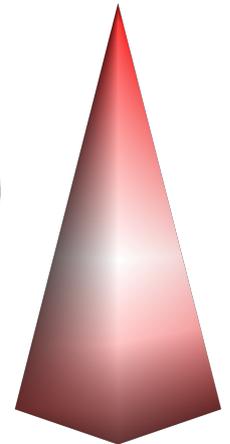
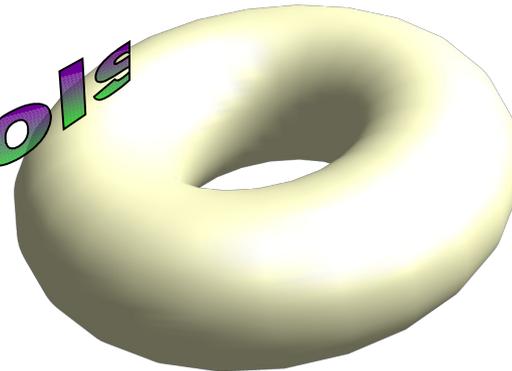
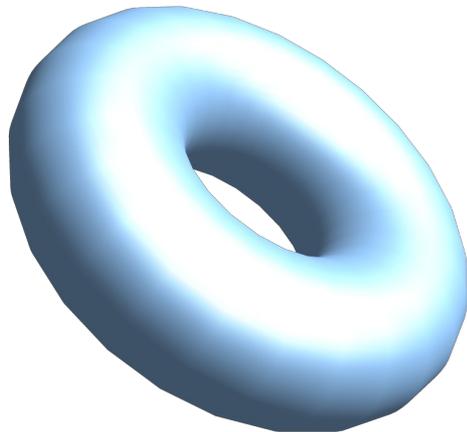


- Will make you a better computer scientist
- UNIX is a building block for many CS concepts
- Open source and stable (no viruses, worms, etc)
- Used in many scientific and industrial settings.
- Huge number of **free** and **well-written** software programs
- Excellent programming environment. Different paradigm.
- Roughly 65% of the world's web servers are Linux/Unix machines running Apache.
- Prerequisite to many other CS courses to follow (Operating Systems, Numerical methods, etc.)

Example: Unix Open Office



Unix Office Tools



Brief History of Unix



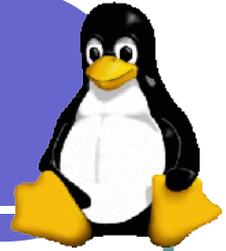
- Ken Thompson and Dennis Ritchie originally developed the earliest versions of Unix at Bell Labs for internal use in the 1970s
 - Simple and elegant
 - Meant for programmers and experts
 - Written in a high-level language instead of assembly language
 - Small portion written in assembly language (kernel)
 - Remaining code written in C on top of the kernel
- <http://www.bell-labs.com/history/unix/>

Brief History of Linux



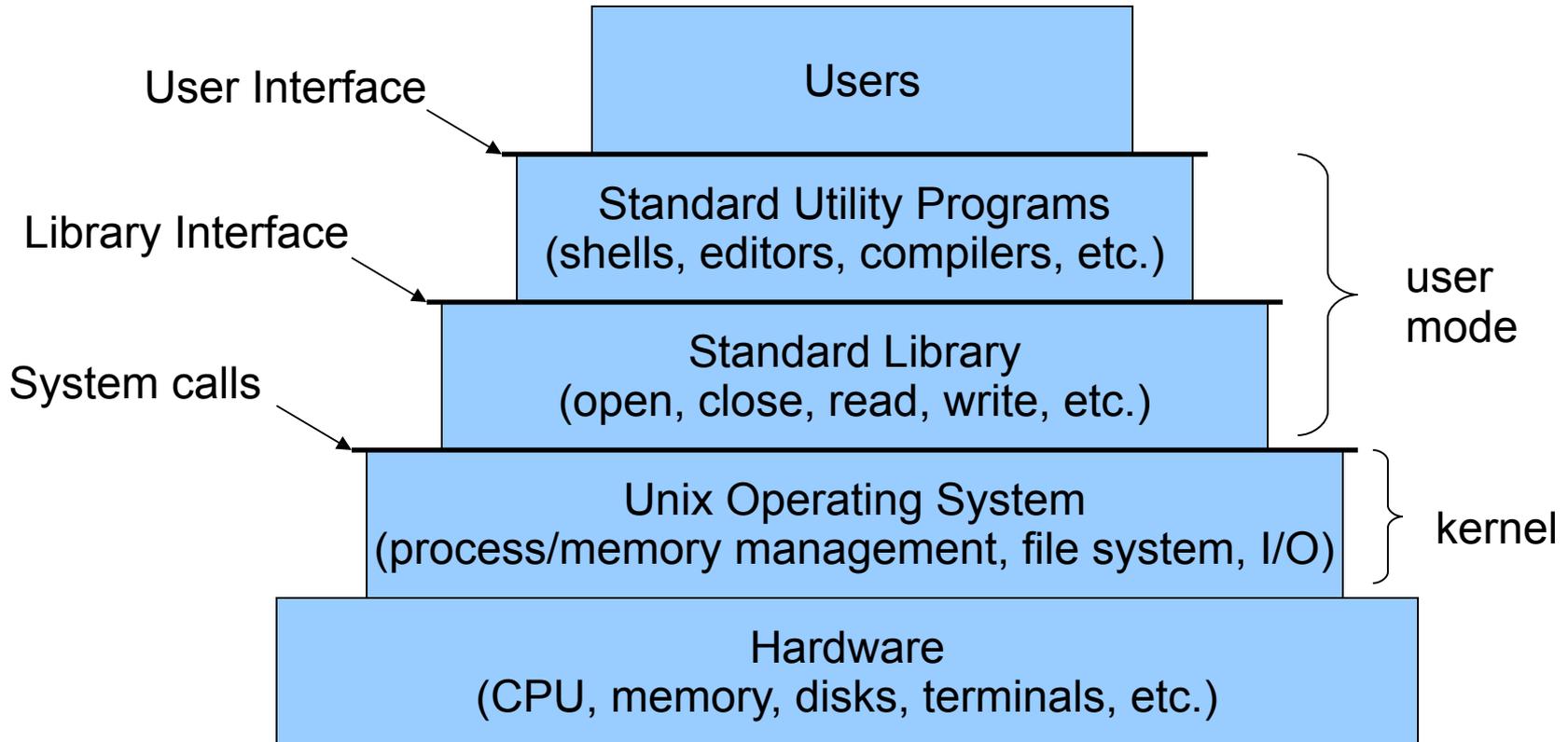
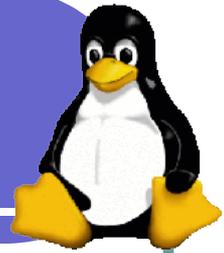
- *Andrew Tanenbaum*, a Dutch professor developed MINIX to teach the inner workings of operating systems to his students
- In 1991 at the University of Helsinki, *Linus Torvalds*, inspired by Richard Stallman's GNU free software project and the knowledge presented in Tanenbaum's operating system, created 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>

Unix Variants (Flavours)



- Two main threads of development
 - Berkeley software distribution (<http://www.bsd.org>)
 - Unix System Laboratories (<http://www.unix.org>)
- Sun: SunOS, Solaris
- SGI: Irix
- FreeBSD, OpenBSD, NetBSD
- Hewlett-Packard: HP-UX
- Apple: OSX (based on BSD)
- Linux (many flavours)

Layers in a Unix-based System

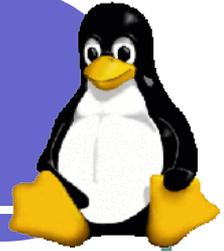


Unix Structure



- 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.

Unix Structure (cont.)



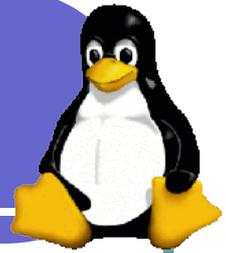
- Many hundreds *utility* programs or *tools* are supplied with the Unix system. These utilities (or commands) support a variety of tasks such as copying files, editing text, performing calculations, and **developing software**.
- This course will introduce a limited number of these utilities and tools, focusing on those that aid in software development.

Getting Started



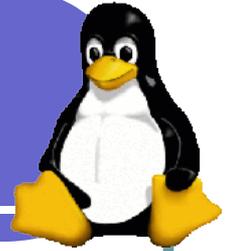
- Logging in to a Unix machine requires an account on that system. Admin = root.
- After logging in, some information about the system will be displayed, followed by a *shell prompt*, where commands may be entered
 - \$
 - %
 - #
 - `username@hostname>`
 - `hostname%`

The Shell



- 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
 - `more textfile`
 - `ls -l /home/onufriev`

Command Syntax



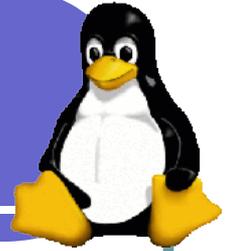
- Commands must be entered exactly
- Be careful! Some commands can be very destructive. (e.g. `rm junk*` vs. `rm junk *`) **NO RECOVERY IN UNIX.**
ask your TA how to safeguard against accidental file removal (alias `rm` to `mv`)
- Syntax: **command** *options* *argument(s)*
- **Options** modify a command's execution
- **Arguments** indicate on what a command should act (often filenames)

Example Commands: ls, cd, mkdir



- `ls -l` // shows content of current directory + file attributes
- `ls -a`
- `ls -la`
- `cd` // move one level up in the directory tree
- `mkdir MYDIRECTORY` // create directory MYDIRECTORY
- `cd MYDIRECTORY`
- `touch myfile` // creates an empty file myfile
- `ls -l myfile`

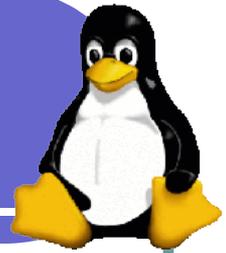
If you don't see a shell prompt...



- 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 by “bg &”)
 - Interrupt the program with CTRL-c (program will usually die)

Absolutely NO MS WINDOWS in this class, BUT you CAN USE PUTTY and WinSCP windows programs to connect to and transfer between a UNIX machine and your windows machine.

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.