Shell characteristics

- Provides command line as an interface between the user and the system
- Is simply a program that starts automatically when you login
- Uses a command *language*
  - Allows programming (shell scripting) within the shell environment
  - Uses variables, loops, conditionals, etc.
- Next week
Various UNIX shells

- sh (Bourne shell)
- ksh (Korn shell)
- csh (C shell)
- tcsh
- bash (Bourne again shell)
- ...
- Differences mostly in scripting details
The Bourne Shell (sh)

- We will be using bash as the standard shell for interaction for this class
- Shell scripting will be done using ksh
- Both language are a superset of the Bourne shell (sh)
- Each language is not necessarily compatible with each other.
Changing your shell

On most UNIX machines:
- `which bash` (note path)
- `chsh`

On the lab machines:
- `which bash` (note path /bin/bash)
- `ypchsh`
Environment variables

- A set of variables the shell uses for certain operations
- Variables have a name and a value
- Current list can be displayed with the `env` command
- A particular variable’s value can be displayed with `echo $<var_name>`
- Some interesting variables: `HOME`, `PATH`, `PS1`, `USER`, `HOSTNAME`, `PWD`
Setting environment variables

- Set a variable with `<name>=<value>`
- Examples:
  - TERM=vt100
  - PS1=myprompt>
  - PS1=$USER@$HOSTNAME:
  - PS1="multiple word prompt>  "
  - PATH=PATH:$HOME
  - DATE=`date`
Aliases

- Aliases are used as shorthand for frequently-used commands
- Syntax: alias <shortcut>=<command>
- Examples:
  - alias ll="ls -lF"
  - alias la="ls -la"
  - alias m=more
  - alias up="cd .."
  - alias prompt="echo $PS1"
Repeating commands

- Use `history` to list the last all previous commands.
- Use `history <k>` to list the last k commands.
- Use `fc -l <m> <n>` to list commands m through n
Some command lines can be very long and complicated - if you make a mistake you don’t want to start all over again.

You can interactively edit the command line in several ways:

- `set -o vi` allows you to use vi commands to edit the command line.
- `set -o emacs` allows you to use emacs commands to edit the command line (such as the up/down arrow keys).
Login scripts

- You don’t want to enter aliases, set environment variables, set up command line editing, etc. each time you log in

- All of these things can be done in a script that is run each time the shell is started

- For bash:
  - `~/.bash_profile` - is read for a login shell
  - `~/.bashrc` - is read for login and other interactive shells
Example .bash_profile (partial)

```bash
if [ -f ~/.bashrc ]; then
  . ~/.bashrc
fi
HOSTNAME=`hostname`; export HOSTNAME
PS1="$USER@$HOSTNAME>"

alias 'll'='ls -l'
alias 'ls'='ls -F'
alias 'rm'='rm -i'

sset -o vi
echo ".bash_profile was read"
```
Each shell (and in fact all programs) automatically open three “files” when they start up

- Standard input (stdin): Usually from the keyboard
- Standard output (stdout): Usually to the terminal
- Standard error (stderr): Usually to the terminal

Programs use these three files when reading (e.g. `scanf()`), writing (e.g. `printf()`), or reporting errors/diagnostics
Redirecting stdout

- Instead of writing to the terminal, you can tell a program to print its output to another file using the > operator

- >> operator is used to append (concatenate) to a file

Examples:
- `man ls > ls_help.txt`
- `Echo $PWD > current_directory`
- `cat file1 >> file2`
Redirecting stdin

- Instead of reading from the terminal, you can tell a program to read from another file using the `<` operator

- Examples:
  - `Mail user@domain.com < message`
  - `interactive_program < command_list`
Pipes and filters

- Pipe: a way to send the output of one command to the input of another
- Filter: a program that takes input and transforms it in some way
  - `wc` - gives a count of words/lines/chars
  - `grep` - searches for lines with a given string
  - `more`
  - `sort` - sorts lines alphabetically or numerically
Examples of filtering

- `ls -la | more`
- `cat file | wc`
- `man bash | grep "history"
- `ls -l | grep "maaguiirr" | wc`
- `who | sort > current_users`