CS-701 Lecture 3

February 6, 2001

Dr. Vickery

Topics

- The Shell
  - Chapters 1, 2, 4, 6, and 8 in Das (!)
- Building an Executable File.
  - The *gcc* compiler driver.
- Unix Commands
  - *man*
  - *make*
  - *tar*
  - *script*

Shell Algorithm

- Display a prompt string
- Read and parse a command line
- Substitute environment variables
- Expand aliases and patterns
- Redirect I/O
- Interpret or Execute command(s)
  - Interpret built-in commands like *cd, setenv, exit,* etc.
  - Create processes to execute files.
- Repeat from the beginning.

Shell Substitutions: Aliases

- If the first word of the command is an alias, the shell replaces the word with the value of the alias.
  - alias l='ls –l' *defines an alias named l*
- Aliases can expand to aliases.
  - alias ls='/usr/bin/ ls –F'
  - alias l='ls –l'
  - Now "l" expands to "/usr/bin/ls –F –l"

Shell Substitutions: Variables

- Any word that starts with a dollar sign is a reference to a shell or environment variable, and the shell substitutes the value of the variable (possibly nothing) for the reference.
  - set PATH=.:$PATH *redefines the environment variable PATH by putting ".:" at the beginning of the current value of the same variable.*

Shell Substitutions: Patterns

- The shell replaces patterns, like *.c, with lists of file names that match the patterns. If there are no matching file names, the pattern itself is passed to the command.
  - The *ls* command prints an error message if the pattern is passed to it because the arguments to *ls* must be the names of files or directories that actually exist.
  - The *echo* command simply prints whatever its arguments are, even if they are patterns that don't match anything.

I/O Redirection

- The shell normally "connects" the user's keyboard to the Standard Input (*stdin*) of the programs it runs, and the terminal window to the Standard Output (*stdout*) before running the program.
- Putting "< *filename*" on the command line tells the shell to connect *stdin* to the file instead of the keyboard.
- Putting "> *filename*" on the command line tells the shell to connect *stdout* to the file instead of the terminal window.
- The program itself doesn't need to be modified in order to read and write files instead of the keyboard and terminal.

Executing a Command

- If the first word of a command (after all substitutions) is the name of a shell built-in, the shell executes the command by executing part of its own code.
- If the command is not a shell built-in, the shell searches the directories listed in the value of the PATH environment variable until it finds an executable file with same name as the command.
  - The shell uses *fork()* to create a process for the command, and *exec()* to get that process to execute the program contained in the file.

Background Commands

- If you type an ampersand (&) at the end of a command, the shell immediately returns to the top of its loop and prints the next prompt instead of waiting for the command to finish before doing so.
  - Background commands must not try to interact with the user because that would interfere with the shell's doing so.