System Programming

UNIX Shell Environments

Shell Characteristics

- Command-line interface between the user and the operating system
- Automatically starts on login, wait for user to type in commands
- Both a command interpreter and a programming language
- Shell script is a text file containing logic for shell interpretation

Shell Interactivity

- Command line parsing
- Environment
- Textual completion
- Aliases
- Command line editing
- Command history
- Configuration

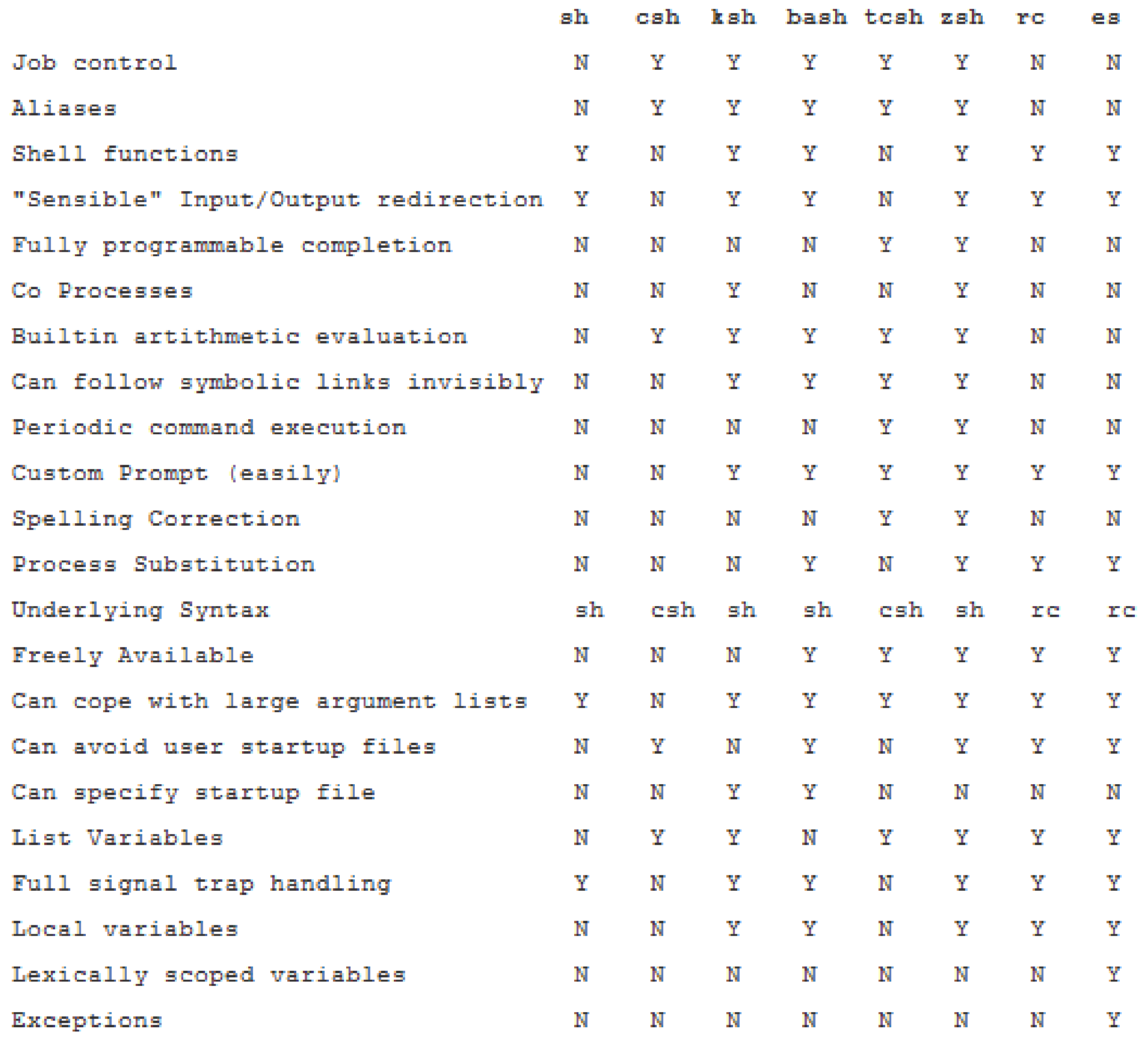
Shell Programming

- Variables
- Control structures
  - Loops and conditionals
- Function definition and invocation
- Shell scripts
- Next chapter

Various Unix Shells

- sh (Bourne shell, original Unix shell)
- ksh (Korn shell)
- csh (C shell, developed at Berkeley)
- tcsh
- bash (Bourne again SHell)
  - Default user shell in Linux
- ...
- Differences mostly in level of interactivity support and scripting details
  - http://www.faqs.org/faqs/unix-faq/shell/shell-differences/

Shell Features



	sh	csh	ksh	bash	tcsh	zsh	rc	es
Job control	N	Y	Y	Y	Y	Y	N	N
Aliases	N	Y	Y	Y	Y	Y	N	N
Shell functions	Y	N	Y	Y	N	Y	Y	Y
"Sensible" Input/Output redirection	Y	N	Y	Y	N	Y	Y	Y
Fully programmable completion	N	N	N	N	Y	Y	N	N
Co Processes	N	N	Y	N	N	Y	N	N
Builtin artithmetic evaluation	N	Y	Y	Y	Y	Y	N	N
Can follow symbolic links invisibly	N	N	Y	Y	Y	Y	N	N
Periodic command execution	N	N	N	N	Y	Y	N	N
Custom Prompt (easily)	N	N	Y	Y	Y	Y	Y	Y
Spelling Correction	N	N	N	N	Y	Y	N	N
Process Substitution	N	N	N	Y	N	Y	Y	Y
Underlying Syntax	sh	csh	sh	sh	csh	sh	rc	rc
Freely Available	N	N	N	Y	Y	Y	Y	Y
Can cope with large argument lists	Y	N	Y	Y	Y	Y	Y	Y
Can avoid user startup files	N	Y	N	Y	N	Y	Y	Y
Can specify startup file	N	N	Y	Y	N	N	N	N
List Variables	N	Y	Y	N	Y	Y	Y	Y
Full signal trap handling	Y	N	Y	Y	N	Y	Y	Y
Local variables	N	N	Y	Y	N	Y	Y	Y
Lexically scoped variables	N	N	N	N	N	N	N	Y
Exceptions	N	N	N	N	N	N	N	Y

Bourne Again SHell (bash)

- Bash is the standard shell for this lecture
- Superset of the Bourne shell (sh)
- Borrows features from sh, csh, tcsh & ksh
- Part of the GNU project

Variables

- Three main types of variables for a running shell
- Local variables
  - Present within current instance of shell
  - Set at command prompt
- Environment variables
  - Available to any child process of shell
- Shell variables
  - Set and required by shell

Shell Variables

- A set of variables the shell uses for certain operations
- Shell variables include
  - local variables
  - environment variables
- Current list of environment variables can be displayed with the `env` command
- Variables have a name and a value
- Send value of `varname` to standard output with `echo $varname`

Environment Variables

- Some interesting variables: `HOME, PWD, PATH, PS1, USER, HOSTNAME`

`$HOME:` home directory (default argument for cd)
  Example: `/home/0607/student`

`$PWD:` current working directory
  Example: `/export/home/staff/usern`

`$PATH:` search path for commands
  Example: `/usr/local/bin:/bin:/usr/bin`

`$PS1:` command prompt (default "`$ `")
  Example: `\u@\h:\w\$`

`$USER:` user name
  Example: `usern`

`$HOSTNAME:` computer hostname
  Example: `ktuce`

Environment Variables

- Other interesting variables: `UID, PPID, RANDOM, HISTFILE, HISTSIZE, MAIL, MAILCHECK, PS2`

`$UID:` current user ID

`$PPID:` process ID of program that invoked the shell

`$RANDOM:` generate a random integer between 0-32767

`$HISTFILE:` file for storing command history

`$HISTSIZE:` the number of commands to be stored

`$MAIL:` file checked by shell for the arrival of mail

`$MAILCHECK:` the number of seconds between checks

`$PS2:` secondary command prompt (default “> ”)

Setting Variables

- Set variable with `varname=value`
- `PS1=$USER@$HOSTNAME:`

  `Change default shell prompt`
- `PS1="bash_prompt> "`
- `PATH=$PATH:$HOME/bin`
  - `Append :$HOME/bin to PATH`
- `PATH=$PATH:~:.`
  - `Append :~:. to PATH`
- ``DATE=`date` or DATE=$(date)``
  - Capture output from `date` command

Textual Completion

- <tab> attempts to complete the current command or filename
- pus<tab> expands to pushd<space>
- pu<tab> gives the alternatives
  - pu pup pushd
- In /etc, entering `ls init<tab>` gives

```
init    init.d    initpipe     inittab
[lecture]$ ls init
```

Aliases

- Aliases are used as shorthand for frequently-used commands
- Syntax: `alias shortcut=command`
- Examples:
  - `alias pu=pushd`
  - `alias po=popd`
  - `alias l= "ls –F -C"`
  - `alias ll="ls –L –l -F"`
  - `alias d=dirs`
  - `alias hide="chmod og-rwx"`
  - `alias unhide="chmod og+r"`

Command History

- Use `history` command to list previously entered commands
- Use `fc -l <m> <n>` to list previously typed commands from m through n
- Use up and down cursor keys to scroll through history list

Editing on the Command Line

- `bash` provides a number of line editing commands
- Default emacs-mode commands
  - `Esc-b` Move back one word
  - `Esc-f` Move forward one word
  - `Ctrl-a` Move to beginning of line
  - `Ctrl-e` Move to end of line
  - `Ctrl-k` Kill text from cursor to end of line
- On the other hand, you can interactively edit the command line in several ways if using `ksh`
  - `set -o vi` allows you to use vi commands to edit the command line
  - `set -o vi-tabcomplete` also lets you complete commands/ filenames by entering a TAB

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

Login Scripts (cont.)

- Startup scripts executed at login
  - `/etc/profile`
  - `~/.bash_profile`
    - `~/.bash_login (if no .bash_profile)`
    - `~/.profile (if neither are present)`
- Script executed after login
  - `~/.bashrc`
- Script executed upon logout
  - `~/.bash_logout`

Example .bash_ profile (partial)

```
# .bash_ profile: executed by bash for login shells
umask 022 (0666 & ~022 = 0644 = rw-r--r--)
# include .bashrc if it exists
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
fi
# Set variables for a warm fuzzy environment
export CVSROOT=~/.cvsroot
export EDITOR=/bin/vi
export PAGER=/usr/bin/less
```

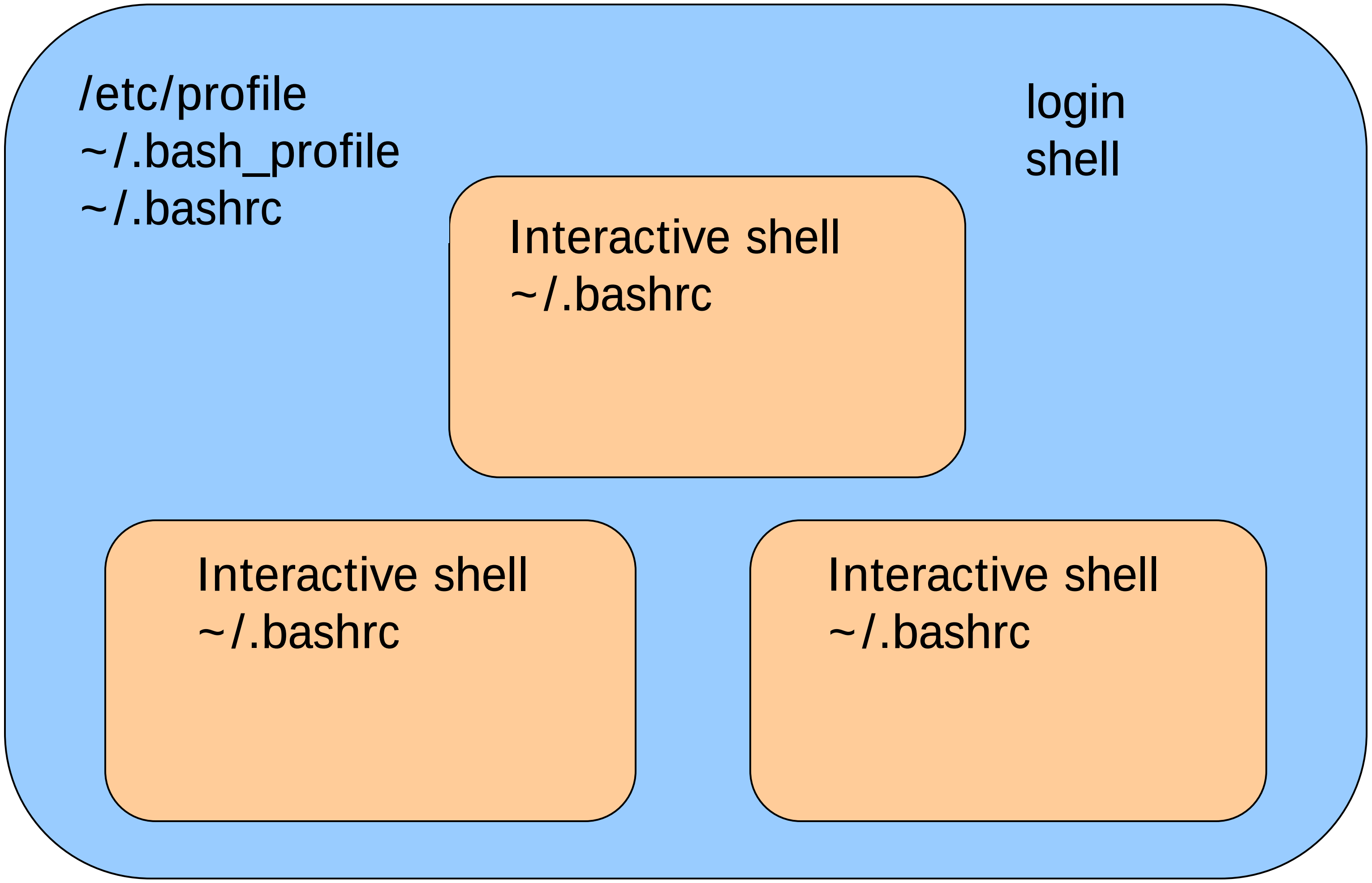
Example .bashrc (partial)

```
# .bashrc
# abbreviations for some common commands
alias bye=logout
alias h=history
alias l='ls –F -C'
alias ll='ls-L –l -F'
alias po=popd
alias pu=pushd
```

Login Scripts (cont.)

- For csh, login shells execute:
  - `~/.profile`
- If ENV is set:
  - That file is executed for each new terminal
  - Example:
    - `ENV=$HOME/.cshrc`
    - `EXPORT ENV (for bash)`

Login and Other Shells


/etc/profile
~/.bash_profile
~/.bashrc
login shell
Interactive shell
~/.bashrc
Interactive shell
~/.bashrc
Interactive shell
~/.bashrc