Lecture # 4.6
Design and Code of
UNIX `shell` utility

Course: Advanced Operating System

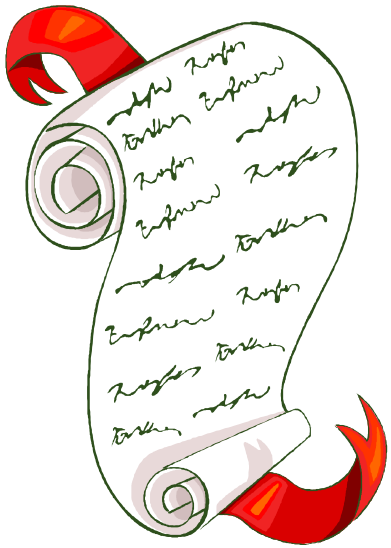
Instructor: Arif Butt

Punjab University College of Information Technology (PUCIT)
University of the Punjab

Source Code files available at: `https://bitbucket.org/arifpucit/spvl-repo/src`
Lecture Slides available at: `http://arifbutt.me`

Agenda

- What is a **shell** program in UNIX
- What does a **shell** program do?
- How does a **shell** do it?
- Writing a **shell** program of our own
- Coding different versions of our own **shell**



What does a SHELL do?

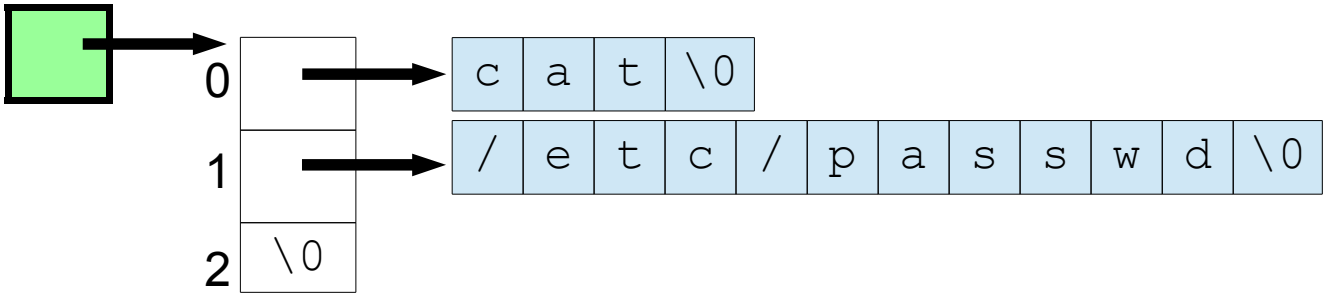
1. Execute commands (Internal or External)
2. Job control
3. Name completion
4. Maintains history of commands
5. Perform I/O redirection
6. Use of pipes (|)
7. Shell variables (environment and user defined)
8. Shell operators (arithmetic, comparison, logical, file)
9. Programming ability (if...else and loops)

How does SHELL do it?

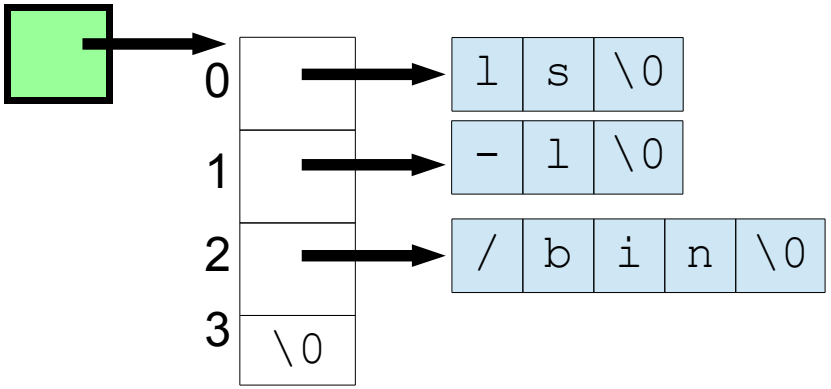
1. Display the prompt
2. Read command line string
3. Tokenizes it
4. If it is a control command handles it
5. If it is an internal command executes its code
6. If it is an external command
   - A Child process is forked and execed with the command
   - Parent process wait for the termination of the child
7. After child terminates, parent process go to step 1

Tokenizing the Command String

```
$ cat /etc/passwd
```




```
$ ls -l /bin
```



myshellv1.c

A version that shows the basic working of a UNIX shell

myshellv2.c

This version should add some internal commands to the shell like `cd`, `exit`, `jobs`, `help` etc.

myshellv3.c

This version should be able to redirect `stdin` and `stdout` for the new processes by using < and > operators. It should also add the functionality of the pipe operator `(|)` as well

myshellv4.c

This version should be able to run multiple commands on a single line separated by semi colons. Moreover, should be able to execute commands in the background if the command is followed by an & operator before the new line character

myshellv5.c

This version should add the feature of maintaining command history and accessing them using the !n command or the up and down arrow keys

myshellv6.c

This version add the functionality of adding an if control structure in your shell. The if control structure works as follows:

- **The shell runs the command that follows the key word if**
- **The shell checks the exit status of the command**
- **An exit status of 0 means success, nonzero means failure**
- **The shell executes commands after the then line if success**
- **The shell executes commands after the else line if failure**
- **The keyword fi marks the end of if block**

myshellv7.c

This version add the feature of creating, accessing and modifying variables within your shell

myshellv8.c

This version should add the feature of file/command name completion on tab key as the bash shell do. The GNU readline library makes this easy

Things To Do

If you have problems visit me in counseling hours. . . .