C S 30B: LINUX SHELL PROGRAMMING

Foothill College Course Outline of Record

Heading	Value
Effective Term:	Summer 2021
Units:	4.5
Hours:	4 lecture, 2 laboratory per week (72 total per quarter)
Advisory:	C S 30A or equivalent.
Degree & Credit Status:	Degree-Applicable Credit Course
Foothill GE:	Non-GE
Transferable:	CSU/UC
Grade Type:	Letter Grade (Request for Pass/No Pass)
Repeatability:	Not Repeatable

Student Learning Outcomes

- A successful student will be able to code basic commands in the BASH programming environment using a structured approach that shows mastery of the write/test/debug cycle. In particular, the student will be able to use arrays, iterative and conditional structures, sorts, regular expressions and nesting in shell scripts.
- A successful student will be able to make us of redirection, pipes, advanced regular expressions, awk, jobs, signals and other advanced scripting techniques.

Description

Linux shell script programming using the Bourne Again shell programming language (bash) and Linux utilities to create practical shell scripts. Topics covered include customizing the environment, running and writing scripts, variables, loops, functions, text processing and debugging.

Course Objectives

The student will be able to:
A. Describe the history, purpose and components of a Linux shell.
B. Create a user account, logon and get information using commands on a Linux system.
C. Understand and customize the bash shell environment by creating aliases and altering environment files.
D. Use a Linux text editor to create a shell script and run scripts effectively from the command line.
E. Write code to redirect input and output to and from the user, files and commands using redirection and pipes.
F. Incorporate essential Linux utilities such as eval, exec, exit and sleep in a program.
G. Describe the different types of variables in the bash environment and explain the properties and uses of each.
H. Define, analyze and code the basic conditional and iterative control structures and explain how they can be nested.
I. Design, implement, test, and debug functions that can be used in scripts, and demonstrate the way parameters are passed in such functions.
J. Declare and initialize an array, then access and sort the array elements.
K. Use regular expressions and bash commands to write code to process text including finding, sorting, comparing and merging.
L. Use advanced regular expressions as well as the sed and awk utilities to perform more advanced text manipulation.
M. Describe methods for handling and using processes, jobs, signals, coroutines, and subshells.
N. Implement code to appropriately handle errors and provide meaningful feedback via exit status and error messages.
O. Use a variety of techniques such as debug functions, debug statements and built-in debugging options to find and fix code errors.

Course Content

A. Shell basics
1. Purpose of a shell
2. History of Linux shells
3. Components of Linux
4. The kernel and subsystems
B. Getting started with Linux
1. How to login and logout
2. User accounts
3. The superuser
4. Account settings and configuration
5. Commands for getting information
6. Stopping a program
C. Customizing the environment
1. .bash_profile file
2. .bash_logout file
3. .bashrc file
4. Aliases
5. Shell prompt
D. Writing and running scripts
1. Text editors
2. Understanding man pages
3. Command line syntax
4. Arguments and options
5. Command line history
6. Command line completion
7. Command line editing
E. Input/Output
1. Standard input
2. Standard output and standard error
3. Redirection
4. Pipes
F. Essential utilities/commands
1. echo and print
2. exit
3. exec
4. eval
5. sleep
G. Variables
1. Bash reserved variables
2. Typed variables
3. Integer variables and arithmetic
4. String variables and quoting
5. Variable scope
6. Exporting variables
H. Program control
1. Logical operators
2. Test command
3. if/else
4. for

5. case
6. select
7. while/until
I. Functions and argument passing
1. Writing functions
2. Calling functions
3. Positional parameters
4. Processing options
J. Arrays
1. Properties
2. Operations
3. Sorting
K. Text processing commands
1. Selecting using grep
2. Basic regular expressions
3. Sorting input
4. Finding files
5. Merging lines
6. Comparing files
7. Eliminating duplicate lines
L. Advanced text manipulation
1. awk
2. sed
3. Advanced regular expressions
M. Process handling
1. Process IDs and job numbers
2. Job control
3. Signals
4. Coroutines
5. Subshells
6. Process substitution
N. Exit and signal handling
1. Signal trapping
2. Exit status
3. Error messages
O. Debugging techniques
1. Syntax vs. logic errors
2. Echo commands
3. Set built in debugging options
4. Debug statements
5. Debug functions

Lab Content

A. Getting started with Linux
1. Create a new user account with user id and password
2. Enter commands such as whois, which and whereis to get basic information
3. Run and stop programs from the command line
4. Customize the environment by altering key system files
B. Writing and running scripts
1. Demonstrate the complete edit-debug-run cycle of a script using a text editor and the command-line environment
2. Use command line arguments and options to run the script
3. Distinguish between syntax errors and logic errors and develop strategies for dealing with each type of error
4. Debug code with the help of error messages and functions to produce a working program
C. Exploring the different variable types through system and user-written files
1. Gain experience in effectively using the text editor to create code with numeric types
2. Gain experience in effectively using the text editor to create code with string types
3. Use the text editor to assist in assigning new values to bash environment and reserved variables
4. Solve syntax and logic problems that arise from typical incorrect use of variables whether pre-defined or user-defined
D. Demonstrating interaction with the user, files and commands
1. Use pipes and redirection to interact with files or other commands
2. Play the role of user and programmer, alternately, to establish a user-interaction plan for a program
3. Evaluate and comment on other students' user-interaction plan
4. Change modes from source code design (editing mode) to end-user interaction (run mode) in your IDE in order to perform Q/A on the program

5. Fix poor interaction behavior by adjusting code and rerunning program until a satisfactory result is achieved
E. Building a script that demonstrates "intelligence" though a combination of control statements
1. Become familiar with selection, loop and nesting to imbue code with correct logic behavior
2. Use structured programming to make control structures maintainable
3. Run the program multiple times to verify that its control statements produce the correct behavior or output under any scenario
4. Fix incorrect logic behavior by adjusting control structures and rerunning program until a satisfactory result is achieved
F. Incorporating functions and utilities in programming projects
1. Gain experience in writing a function
2. Use a previously written function/utility in a client program
3. Refine functions by adding or changing their definitions and observe the result
4. Use positional parameters and processing options to change the outcome of running a function
5. Deduce the impact of a function's or utility's design on the programs that invoke it
G. Processing and manipulating text
1. Use commands, regular expressions, and utilities such as sed and awk to process and manipulate text
2. Use the man pages to understand the purpose and use of various text processing commands
3. Use debugging techniques to successfully integrate awk and sed functions into user written code
H. Handling processes
1. Use a text editor to write code involving both system and user processes
2. Incorporate commands to control jobs, handle signals, run coroutines and start child processes
3. Show results by using commands to print information on processes
4. Use debugging techniques to solve problems that arise during the testing of the program

Special Facilities and/or Equipment

A. Access to a computer laboratory with the Linux operating system.
B. A website or course management system with an assignment posting component (through which all lab assignments are to be submitted) and a forum component (where students can discuss course material and receive help from the instructor). This applies to all sections, including on-campus (i.e., face-to-face) offerings.
C. When taught via Foothill Global Access on the Internet, the college will provide a fully functional and maintained course management system through which the instructor and students can interact.

D. When taught via Foothill Global Access on the Internet, students must have currently existing email accounts and ongoing access to computers with internet capabilities.

Method(s) of Evaluation

Methods of Evaluation may include but are not limited to the following:

Tests and quizzes
Written laboratory assignments which include source code, sample runs and documentation
Final examination

Method(s) of Instruction

Methods of Instruction may include but are not limited to the following:

Lectures which include motivation for syntax and use of shell programming language and shell scripting concepts, example programs, and analysis of these programs
Online labs (for all sections, including those meeting face-to-face/on-campus), consisting of:
1. An assignment webpage located on a college-hosted course management system or other department-approved internet environment. Here, the students will review the specification of each lab assignment and submit their completed lab work
2. A discussion webpage located on a college-hosted course management system or other department-approved internet environment. Here, students can request assistance from the instructor and interact publicly with other class members
Detailed review of written assignments which includes model solutions and specific comments on the student submissions
In person or online discussion which engages students and instructor in an ongoing dialog pertaining to all aspects of designing, implementing and analyzing shell scripts
When course is taught fully online:
1. Instructor-authored lecture materials, handouts, syllabus, assignments, tests, and other relevant course material will be delivered through a college-hosted course management system or other department-approved internet environment
2. Additional instructional guidelines for this course are listed in the attached addendum of CS department online practices

Representative Text(s) and Other Materials

Blum, Richard, and Christine Bresnahan. Linux Command Line and Shell Scripting Bible, 3rd ed.. 2015.

Tammer, Sebastiaan. Learn Linux Shell Scripting – Fundamentals of Bash 4.4, 3rd ed.. 2018.

Types and/or Examples of Required Reading, Writing, and Outside of Class Assignments

A. Reading
1. Textbook assigned reading averaging 30 pages per week.
2. Reading the supplied handouts and modules averaging 10 pages per week.
3. Reading online resources as directed by instructor though links pertinent to bash shell scripting.
4. Reading library and reference material directed by instructor through course handouts.
B. Writing
1. Writing technical prose documentation that supports and describes the programs that are submitted for grades.

Discipline(s)

Computer Science