I. GENERAL DESCRIPTION
A. Date March 2010
B. Department Computer Science
C. Course Number CS 160B
D. Course Title Unix/Linux Shell Scripting
E. Course Outline Preparer Peter Wood, Greg Boyd, Charlie Metzler
F. Department Chairperson Constance Conner
G. Dean Tom Boegel

II. COURSE SPECIFICS
A. Hours Lecture-2, Laboratory-2
B. Units 2
C. Prerequisites CS 160A or demonstration of CS 160A exit skills
   Corequisites none
   Advisories none
D. Course Justification System administration jobs require shell scripting skills. This course will bring a student’s shell scripting skills to a level that will help prepare them for CS 260A and CNIT 270.
E. Field Trips No
F. Method of Grading Letter, Pass/No Pass
G. Repeatability 0

III. CATALOG DESCRIPTION
Analyze, design, write, test, and debug Unix shell scripts. Students learn basic shell scripting techniques and develop scripting skills needed for Unix/Linux System Administration courses. The bash shell is used.

IV. MAJOR LEARNING OUTCOMES
Upon completion of this course, the student will be able to:
A. Identify and describe the components of a well-written and documented shell script.
B. Create sequences of Unix commands that use regular expressions and standard Unix text filters to filter text.
C. Use conditional statements to control the execution of shell scripts.
D. Write shell scripts to perform repetitive tasks using while and for loops.
E. Analyze existing shell scripts.
F. Apply redirection in a script to read from and write to files, to interact with the user, and to control standard output and standard error.
G. Design and implement shell functions with limited side-effects for use in existing code and shell function libraries.
H. Use arithmetic and command substitution and quoting rules to create complex commands and to assign data to variables.
I. Analyze common problems, identify solutions, and create scripts to solve them.
V. CONTENT
   A. Review of Basic Unix Concepts
      1. Shell differences
      2. Script basics
      3. Working with files and directories
      4. Using find to locate files and directories by name pattern and type
      5. Manipulating file attributes: types, owners, groups and permissions.
      6. Using text filters such as head, tail, cut, tr, sort, uniq, grep, sed and awk.
      7. Basic regular expressions
   B. Shell Programming
      1. Script Basics
      2. Variables
         a. local
         b. environment
      3. Substitution
         a. shell wildcards
         b. variable
         c. command, including backquote and $(...) forms
         d. arithmetic, including expr and $(...) forms
      4. Quoting
         a. backslashes
         b. single quotes
         c. double quotes
         d. quoting rules and situations
      5. Flow Control
         a. the test command
         b. the if statement
         c. the case statement
         d. && and ||
      6. Loops
         a. the for loop
         b. the while loop
         c. break and continue
      7. Documentation
         a. Comments
         b. Indentation
      8. Parameters
         a. special variables ($#, $*, $0, $1 ...)
         b. options and arguments
         c. using ‘set’ to set command line arguments
         d. option parsing in shell scripts
      9. Input/Output file descriptors and redirection in scripts
         a. standard input, standard output, and standard error
         b. output redirection, including: >, 2>, >> and 2>>
         c. input redirection, including < and <<
         d. combining output streams using: 2>&1, >&2
e. using the read command to read data from files and interactively from the user

10. Functions
   a. creating and using functions
   b. scoping issues

11. Advanced Topics
   a. forcing re-evaluation of the command using eval
   b. the null command (\)
   c. using type to identify the location of a standard command
   d. using sleep to implement delays
   e. use of "safe" temporary files by generating names using $$ or mktemp
   f. creation and use of variables containing file paths.
   g. sourcing using the . operator

C. Advanced Topics
   1. Debugging
      a. shell tracing
      b. adding conditional debugging code
   2. Function Libraries
      a. creating a library of functions
      b. useful functions
   3. Steps in designing a script
   4. Techniques for increasing portability

VI. INSTRUCTIONAL METHODOLOGY

A. Assignments
   1. In-class Assignments
      a. Group coding of example shell scripting problems
      b. Analyzing existing shell scripts
   2. Out-of-class Assignments
      a. Creating shell scripts that use text filters to filter the output of Unix commands, analyze the results, and report findings to the user
      b. Combining redirection, conditionals, and looping in a script.
      c. Writing several functions and including them in a script
      d. Writing scripts that use and analyze arguments to perform their tasks as do standard Unix commands.

B. Evaluation
   1. Student assignments as described above that measure the student’s ability to understand Unix shell scripting, filter text, and use Unix utilities, control structures, redirection, and functions to effectively analyze problems and solve them with shell scripts.
   2. Periodic tests including a comprehensive in-class written final that measure the student’s ability to understand concepts, such as: shell variables, command and arithmetic substitution, quoting rules, how to use regular expressions and text filters to filter text, shell control structures, functions, redirection, problem solving, and portability issues.
City College of San Francisco  
Course Outline of Record

C. Textbooks and Other Instructional Materials
3. Class handouts such as practice exercises, assignments, or review notes.
4. Remote access to a Unix/Linux server.

VII. TITLE 5 CLASSIFICATION
VIII. CREDIT/DEGREE APPLICABLE (meets all standards of Title 5. Section 55002(a)).