UNIX

Duration – UNIX- 6 Days

UNIX Operating System

DURATION 3 DAYS

OVERVIEW

- Objectives
- Introduction to OS
- UNIX History
- UNIX Principles
- Running Commands
- Some Simple Commands
- Getting Help
- The whatis Command
- The --help Option
- Reading Usage Summaries
- The man Command
- Navigating man Pages
- Hands-on lab: Getting Help with Commands

BROWSING THE FILESYSTEM

- Objectives
- Some Important Directories
- Other Important Directories
- Current Working Directory
- File and Directory Names
- Absolute Pathnames
- Relative Pathnames
- Changing Directories
- Listing Directory Contents
- Copying Files and Directories
- Copying Files and Directories: The Destination
- Moving and Renaming Files and Directories
- Moving and Renaming Files and Directories: The Destination
- Creating and Removing Files
- Creating and Removing Directories
- Determining File Content
- Viewing an Entire Text File
- Viewing Text Page by Page
- Hands-on lab: Browsing the Filesystem

STANDARD I/O AND PIPES

- Objectives
- Standard Input and Output
- Redirecting Input and Output
- Redirecting Output
- Redirecting Standard Output
- Overwriting vs Appending
- Redirecting Standard Error
- Redirecting Both Standard Output and Error
- Redirecting Input
- Using Pipes To Connect Processes
- Useful Pipe Targets
- tee
- Hands-on lab: Standard I/O and Pipes

USERS, GROUPS, AND PERMISSIONS

- Objectives
- The Unix Security Model
- Users
- Groups
- The root user
- Unix File Security
- Permission Types

- Examining Permissions
- Interpreting Permissions
- Examining Directories
- Unix Process Security
- Changing Permissions - Symbolic Method
- Changing Permissions - Numeric Method
- Hands-on lab: File Permissions

VI AND VIM EDITOR BASICS

- Objectives
- Overview of vi
- Starting vi
- Three Modes of vi
- Cursor Movement
- Cursor Movement
- Entering Insert Mode
- Leaving Insert Mode: Esc
- Change, Delete, and Yank
- Put (paste)
- Undoing Changes
- Searching for Text
- Command-Mode Tricks
- Saving and Exiting: ex mode
- Hands-on lab: vi Editor Basics

THE UNIXFILESYSTEM IN-DEPTH

- Objectives
- Partitions and Filesystems
- Inodes
- Directories
- Inodes and Directories
- cp and inodes
- mv and inodes
- rm and inodes
- Symbolic (or Soft) Links
- Hard Links
- The Seven Fundamental Filetypes
- Checking Free Space
- Why Archive Files?
- Creating an Archive
- Inspecting Archives
- Extracting an Archive
- Why Use File Compression?
- Compression Utilities
- Using Compression
- Compressing Archives
- Hands-on lab: The UnixFilesystem

ADVANCED TOPICS IN USERS, GROUPS AND PERMISSIONS

- Objectives
- User and Group ID Numbers
- /etc/passwd, /etc/shadow, and /etc/group files
- System Users and Groups
- Changing Your Identity
- User Information Commands
- Default Permissions
- Special Permissions
- Special Permissions for Executables
- Special Permissions for Directories
- Hands-on lab: Switching Users and Setting a Umask

INTRODUCTION TO STRING PROCESSING

- Objectives
- head, tail, tail, wc (word count), sort, uniq, cut&Other String Processing Tools
- Version Comparison with diff
- Spell Checking with aspell
- Hands-on lab: Introduction to String Processing

STRING PROCESSING WITH REGULAR EXPRESSIONS

- Objectives

- Pattern Matching with Regular Expressions
- Wildcard Characters
- Character Classes
- Modifiers
- Anchors
- The | Operator
- regex Combinations
- Regular Expressions - Examples
- Quote your regex´s!
- grep, sed, Using sed
- less and vi
- Hands-on lab: String Processing with Regular Expressions

FINDING AND PROCESSING FILES

- Objectives
- find
- Basic find Examples
- find and Logical Operators
- find and Permissions
- find and Numeric Criteria
- find and Access Times
- Executing Commands with find
- find Execution Examples
- The Gnome Search Tool
- Hands-on lab: Finding and Processing Files

INVESTIGATING AND MANAGING PROCESSES

- Objectives
- What is a Process?
- How Processes Are Created
- Process Ancestry
- Process States
- Viewing Processes
- Sending Signals to Processes
- Terminating Processes
- Altering Process Scheduling Priority
- Altering Process Scheduling Priority (continued)
- Interactive Process Management Tools
- Running a Process in the Foreground
- Running a Process in the Background
- Suspending a Process
- Listing Background and Suspended Jobs
- Resuming Suspended Jobs
- Compound Commands
- Scheduling a Process To Execute Later
- Scheduling Periodic Processes
- Using cron
- Crontab File Format
- Hands-on lab: Process Controls

UNIX SHELL SCRIPTING

Duration 3 Days

UNIX SHELLS AND SHELL SCRIPTS

- Describe the role of shells in the UNIX environment
- Describe the standard shells
- Define the components of a shell script
- Write a simple shell script

WRITING AND DEBUGGING SCRIPTS

- Start a script with #!
- Put comments in a script
- Change permissions on a script
- Execute a script
- Debug a script

THE SHELL ENVIRONMENT

- Use Bourne and Korn shell variables
- Assign values to shell variables
- Display the value of shell variables
- Make variables available to subprocesses using the export statement
- Display the value of environment variables
- Unset shell and environment variables
- Customize the user environment using the .profile file
- Perform arithmetic operations
- Create and use aliases
- Display aliases and the values assigned to them
- Define the built-in aliases
- Customize the Bourne and Korn shell environments
- Use the tilde expansion and command substitution features of the Korn shell

CONDITIONALS

- Use the exit status of a command as conditional control
- Use the "if" statement to test a condition
- Pass values using command-line arguments (positional parameters) into a script
- Create USAGE messages
- Place parameters on the command line
- Use conditional if, then, elif, else, and fi constructs
- Use exit, let, and test statements ([[ ]], " ")
- Apply the &&, ||, and ! Boolean logic operators
- Use the case statement

INTERACTIVE SCRIPTS

- Use the print and echo commands to display text
- Use the read command to interactively assign data to a shell variable
- Read user input into one or more variables, using one read statement
- Use special characters, with print and echo, to make the displayed text more user friendly
- Create a "here" document
- Use file descriptors to read from and write to multiple files

LOOPS

- Write scripts that use for, while, and until loops
- Write a script using the select statement
- Describe when to use loops within a script
- Generate argument lists using command, variable, and file-name substitution

THE SED EDITOR

- Use the sed editor to perform noninteractive editing tasks
- Use regular expression characters with the sed command

THE AWK PROGRAMMING LANGUAGE

- Use awk commands from the command line
- Write simple awk programs to generate data reports from text files
- Write simple awk programs to generate numeric and text reports from text files

ADVANCED VARIABLES, PARAMETERS, AND ARGUMENT LISTS

- Declare strings, integers, and array variables
- Manipulate string variables
- Change the values of the positional parameters using the set statement within a script
- Use Korn shell arrays
- Set default values for parameters
- Use the Korn shell built-in let, print, set, and typeset statements

FUNCTIONS

- Create user-defined functions in a shell script
- Create, invoke, and display functions from the command line
- Pass arguments into a function
- Call functions from special (function) files that are saved in one or more function directories
- Describe where functions are available for use

TRAPS

- Describe how the trap statement works
- Include trap statements in a script
- Use the trap statement to catch signals and handle errors

sbmadake@gmail.com 9922292223

sbmadake@gmail.com 9922292223