National University of computer and emerging sciences

Operating Systems Lab (CL-2006)

Semester: Spring 2026
Section: BCS-4H
Course Instructor: Mr. Abdullah Shaikh

LAB # 03 – TASKS

Getting Started with Shell Scripting

Submission Instructions:

1. Perform all the assigned tasks carefully.
2. Prepare your solutions in a Microsoft Word document, clearly writing the solution of each problem.
3. Attach relevant screenshots for every task as proof of execution.
4. Ensure that your document is properly formatted (clear headings, proper spacing, and alignment).
5. After completing the Word document, convert it into a PDF file.
6. Upload the PDF file on Google Classroom (GCR) within the given deadline.
7. No submission will be accepted after the deadline.
8. Before uploading, rename your PDF file using the following format:
   OS Lab 03 – Your Name – Your Roll Number
   Example: OS Lab 03 – Abdullah Shaikh – 25K-7826

Task 01:

Explain what a Linux shell is and list at least three types of shells available in Linux. Check the shell type on your system using a command. Then, create your first shell script that prints "Hello, Linux!" to the terminal. Include a proper shebang line at the top of your script and add comments explaining each step. Run your script and capture the output.

Task 02:

Write a shell script that demonstrates the use of variables. Declare at least two string variables and one integer variable, modify their values, and display them. Next, create an array containing at least five elements (e.g., names or numbers). Access and display individual elements, display all elements, and add a new element to the array. Loop through the array and print each element individually.

Task 03:

Write a shell script that performs string operations. Ask the user to input their name and favorite color using the read command. Concatenate the input to display a message like "Hello [Name], your favorite color is [Color]". Perform additional operations such as finding the length of the name, extracting the first three letters of the color, and comparing the color to a predefined value.

Task 04:

Write a shell script that takes two numbers from the user as input and performs arithmetic operations (addition, subtraction, multiplication, division, modulus). Use if-else and elif statements to check and print whether the first number is greater than, less than, or equal to the second number.

Task 05:

Create a menu-driven shell script using a case statement. The script should display a menu with four options (e.g., Display Date, List Files, Show Current Directory, Exit). Accept the user's choice and perform the corresponding action. Use logical operators to validate the user input and display an error message if an invalid option is selected.

Task 06:

Write a shell script using a for loop to print numbers from 1 to 10. Modify the script to read a list of words from a file and display them using a for loop. Then, create a while loop that counts down from 10 to 1 and an until loop that prints numbers from 1 to 5. Finally, write an infinite loop that prints "Press Ctrl+C to stop" every 2 seconds.

Task 07:

Write a shell script that reads a text file line by line using a while loop. For each line, check if it contains the word "Linux". If it does, print the line with a message "Found Linux: [line]". Count the total number of lines containing the word "Linux" and display the count at the end.

Task 08:

Write a shell script that defines a function to calculate the factorial of a number. Accept the number as a command-line argument when running the script. Call the function with the argument and display the result. Modify the script to accept multiple numbers as arguments and calculate the factorial for each number using the function.

Task 09:

You are tasked with creating a bash script that renames multiple files in a directory according to a specified naming convention. The script should:

a. Accept two arguments: the directory path containing the files and the new file name pattern.

b. Rename each file in the directory by appending a sequential number to the new file name pattern (e.g., `file1.txt`, `file2.txt`, etc.).

c. Preserve the original file extension during the renaming process.

d. Provide feedback to the user about the renaming process, including any errors encountered.

Task 10:

Develop a bash script to automate directory cleanup tasks by removing old files and directories. The script should:

a. Accept a directory path as an argument.

b. Identify and delete files older than a specified number of days.

c. Recursively remove empty directories within the specified directory.

d. Provide feedback to the user about the cleanup process, including the number of files and

directories removed.

THE END!