**Experiment No.: 2**

**Title:** **Shell Programming and system calls**

|  |
| --- |
| **Aim and Objective of the Experiment:** |
| To study the shell script and write the program using shell. |

|  |
| --- |
| **COs to be achieved:** |
| **CO1:** Explain the fundamental concepts of operating system  **CO5:** Explain basic features of Unix and Mobile OS |

|  |
| --- |
| **Theory:** |
| The shell provides you with an interface to the UNIX system. It gathers input from you and executes programs based on that input. When a program finishes executing, it displays that program's output.  **Shell Scripts:**  The basic concept of a shell script is a list of commands, which are listed in the order of execution. A good shell script will have comments, preceded by a pound sign, #, describing the steps.  **Steps to create a Shell Script:**   1. create a file using any text editor say vi, gedit, nano etc. 2. $ vi filename 3. Insert the script/ commands in file and save the file to execute the file we need to give execute permission to the file 4. $ chmod 775 filename 5. Now execute the above file using any of following methods: 6. $ sh filename 7. OR 8. $ ./filename   NOTE: Before adding anything to your script, you need to alert the system that a shell script is being started. This is done using the shebang construct. For example −  #!/bin/sh. |
| **Implementation details/Circuit Diagram/ Block Diagram:** |
| 1. **Write a shell Script that accepts two file names as command line arguments and compare two file contents and check whether contents are same or not. If they are same, then delete second file.**   #!/bin/bash  if [ $# -ne 2 ]; then  echo "Usage: $0 file1 file2"  exit 1  fi  file1=$1  file2=$2  if cmp -s "$file1" "$file2"; then  echo "The files have the same contents."  echo "Deleting $file2..."  rm "$file2"  else  echo "The files have different contents."  fi     1. **Write a shell script that accepts integer and find the factorial of the number.**   #!/bin/bash  echo "Enter a number: "  read number  fact=1  while [ $number -gt 1 ]  do  fact=$((fact \* number))  number=$((number - 1))  done  echo $fact    **Program for System Call:**   1. **Write a Program for creating process using System call (e.g. fork ()) Create a child process. Display the details about that process using getpid and getppid functions. In a child process, Open the file using file system calls and read the contents and display.**   #include <stdio.h> #include <stdlib.h> #include <unistd.h> #include <fcntl.h> #include <sys/types.h> #include <sys/stat.h>  int main() {     pid\_t pid;     pid = fork();  // Create a child process      if (pid < 0) {         // Error in forking         perror("Fork failed");         exit(1);     }      if (pid == 0) {         // Child process         printf("Child Process:\n");         printf("PID: %d\n", getpid());         printf("PPID: %d\n", getppid());          // Open a file and read contents         int file = open("example.txt", O\_RDONLY); // Change "example.txt" to a valid file path         if (file < 0) {             perror("Error opening file");             exit(1);         }          char buffer[256];         int bytesRead = read(file, buffer, sizeof(buffer) - 1);         if (bytesRead < 0) {             perror("Error reading file");             exit(1);         }          buffer[bytesRead] = '\0'; // Null-terminate the buffer         printf("Contents of the file:\n%s\n", buffer);          close(file); // Close the file         exit(0);     } else {         // Parent process         printf("Parent Process:\n");         printf("PID: %d\n", getpid());         printf("PPID: %d\n", getppid());          // Wait for the child process to finish         wait(NULL);     }      return 0; } |

|  |
| --- |
| **Post Lab Subjective/Objective type Questions:** |
| 1. **What are the different types of commonly used shells on a typical linux system?**  * Bash (Bourne Again Shell):   The default shell for most Linux distributions.  Compatible with the Bourne Shell (sh) but with additional features like command history, scripting enhancements, and tab completion.   * Sh (Bourne Shell):   The original Unix shell, widely used for scripting in early Unix systems.  Lightweight and simple but lacks advanced features compared to Bash.   * Zsh (Z Shell):   Known for customization, scripting, and user-friendly features like better auto-completion and improved history management.  Popular for its plugins and themes, especially with the Oh My Zsh framework.   * Ksh (Korn Shell):   Combines features of the Bourne Shell and C Shell.  Known for its scripting capabilities and performance.   1. **List the advantages and disadvantages of shell scripting.**   Advantages:   * Automation: Ideal for automating repetitive tasks like file management, backups, and system maintenance. * Efficiency: Scripts can execute commands quickly and handle complex tasks with minimal user interaction. * Portability: Shell scripts can run on any Unix-like system with minimal or no modifications.   Disadvantages:   * Performance: Shell scripts are slower compared to compiled programs as they are interpreted. * Limited Debugging Tools: Debugging shell scripts can be challenging due to less sophisticated tools compared to programming languages. * Error-Prone: Minor syntax errors can lead to unexpected behavior or script failure. * Lacks strict typing, which can result in bugs. |

|  |
| --- |
| **Conclusion:** |
| In this experiment, we studied the basics of shell scripting and wrote a program using shell commands. This helped in understanding command execution, and scripting logic in a Linux environment. |

|  |
| --- |
| **Signature of faculty in-charge with Date:** |