Assignment - 1
CS342: Operating System Lab

General Instruction

- Assignments should be completed and evaluated in the Lab session.
- Markings will be based on the correctness and soundness of the outputs. Marks will be deducted in case of plagiarism.
- Proper indentation & appropriate comments (if necessary) are mandatory in the code.
- Write the code in .sh file

Welcome to your first lab assignment for Operating Systems! In this assignment, you'll be working on essential scripting tasks that will help you understand and manage various aspects of the system. Enjoy exploring and applying these foundational skills.

1. In this task, you need to set up a project directory structure for efficient management. Start by creating a directory named Project in your home directory, and within it, set up three subdirectories: src for source code, docs for documentation, and bin for binaries. In the src directory, create three files: main.c, utils.c, and Makefile. Move the Makefile to the bin directory, rename main.c to program.c, and copy program.c to the docs directory. Finally, verify the structure by listing the contents of the Project directory and its subdirectories using ls -R /Project.
2. Write a shell script that prompts the user to enter a directory path, validates if the path is a valid directory, and then iterates through all files in the directory to delete any that are empty. The script should log the names of the deleted files in a file named deletedfiles.txt, print the list of deleted files, and if no files are deleted, print a message indicating that no empty files were found. After printing, the script should clean up by removing the deletedfiles.txt file. Save the script to a file (e.g., deleteEmptyFilesWithLog.sh), make it executable, and run it to test its functionality.
3. Write a shell script that takes a directory path as an argument and calculates the total size of all files in that directory, excluding subdirectories. The script should print the total size in a human-readable format, such as KB or MB. If the specified directory does not exist, the script should print an appropriate error message. Ensure that the output clearly indicates the size in the correct units.
4. Write a script named resourceReport.sh that performs the following tasks: Generate a report of CPU and memory usage for all running processes, save this report to a file named resourceReport.txt, and ensure that the report includes the process name, PID, CPU usage, and memory usage. The report should be sorted by CPU usage in descending order. The script should first create or clear the resourceReport.txt file, retrieve and format the process information, and finally print a message to the console indicating that the report has been saved successfully.
5. Write a script named fileSummary.sh that performs the following tasks: First, print the disk usage of the root directory (/). Second, list all files in the current directory. Lastly, display the free memory available on the system. Ensure that the script executes these tasks sequentially and provides clear, informative output for each of these operations.