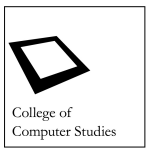

De La Salle University
College of Computer Studies
Software Technology Department

INTROOS: Introduction to Operating Systems
Machine Project Specifications

Expected Learning Outcomes:

LO1. Apply a solid foundation on the understanding of processes and scheduling
LO2. Implement a file management or process management module

Instructions:

1. As a group, choose whether to implement a Process Manager or a File Manager. Also, they should "recreate" the project following the interface/styles of their preferred chosen OS not make it in the environment of their preferred OS. (Example: If you chose Windows File Manager, it should look like Windows 8 Windows Explorer window; If you chose Mac Process manager, it should look like the Mac OS X Activity Montor).
2. Process Manager Specifications
    a. The Process Manager should be implemented with a graphical user interface, built on the platform of choice by the group.
    b. The GUI of the Process Manager should display the following:
        i. All running processes
        ii. The Process ID, CPU, Disk and Process utilization of each process
        iii. (Extra credit) It should show the change in the lines/bars on the information on the utilization of each process.
        iv. (Extra credit) Line and Graphs should update and change in real-time (without the need to "refresh" the stats)
    c. The Process Manager should be able to run a C program that will be provided by the instructor during MP demo day. There should be an option to "Run New Process" and upon loading, should open a console window where the C program will be executed and displayed.
    d. The said loaded process and its information should also be displayed in the Process Manager.
    e. The Process manager should have a feature to "terminate" the process that will be provided by the instructor.
    f. (Extra Credit) The Process manager should be able to duplicate the process loaded by the instructor.
    g. (Even extra credit) The Process manager should be able to duplicate and fork these processes.
3. File Manager Specifications
    a. The File Manager should be implemented with a command-line interface, built on the platform of choice by the group.
    b. The CLI of the File Manager should display the following:
        i. The current files in the active directory.
        ii. The information of each of the files in the active directory like Data Created, Date modified, owner, and file size (extra credit, file size should be measured in the most applicable unit: for example: if file is 1024 KB, it should display instead 1 MB).
        iii. (Extra Credit) It should display how much space in the disk has been occupied and how much are unused.
        iv. (Extra credit) It should display the percentage of space in the hard disk that is being consumed by each file.
    c. The File Manager should be able to:
        i. Edit the file name
        ii. Edit the extension
        iii. Move the file (to a new directory)
        iv. Copy the file (to a new directory only)
        v. Delete the file
    d. The File manager should also be able to run a certain C program that will be provided by the instructor. The said C file will be provided and pasted by the instructor to a desired

directory. From the File Manager, it should be able to run this C program and return to the console once the program has ended running/executing.

e. (Extra Credit) The File manager should be able to edit the contents of the file inside the console itself (without opening an external file). This is similar to the VI used by Unix-based systems (learn more from here: https://www.ccsf.edu/Pub/Fac/vi.html) . Upon editing, the updated date modified and file size should be seen when the contents of the directory are being displayed.

f. (Extra Credit) The File manager should be able to duplicate a file in the same directory and the name would immediately have a (1) or (2) or "Copy of" prefix.

4. Both Process and File Managers when presented to class should be in a packaged version (.exe, .jar).
5. Extensive code documentation (like in Java using Javadoc) should be submitted soon.
6. The following are optional and will be considered extra credit if and only if the required components above have been successfully-presented (F for File, P for Process/ Manager):
    a. (FM) Being able to see a file in a networked location.
    b. (FM) Being able to rename, edit, move and delete files in a networked location.
    c. (FM) Being able to copy a file from a networked location to your local directory (and vice versa)
    d. (FM) Being able to run a certain C file in a networked location.
    e. (PM) Being able to view the processes of a remote computer (possibly connected thru LAN or WLAN)
    f. (PM) Being able to terminate a process of a remote computer.
    g. (PM) Being able to run a certain C process located in a remote computer, the process should be ran on the remote computer.
    h. (PM) Being able to run the same C process and fork them on both computers (the main computer and the remote computer)
    i. Any other extra feature that the instructor can deem acceptable
7. The MP demo day will be on April 8 from 9am to 12noon.
8. All other revisions will be presented on April 12, time and venue to be announced.
9. All deliverables should be packaged along with documentations and other supporting files used, compressed in a .zip/.rar file and submitted by April 13 (Wednesday) 11:59pm following the given specifications:
    a. Subject: [INTROOS] MP <OS Name>
    (example: [INTROOS] MP WinDose)
    b. Body: Names of each members, surname first, written in alphabetical order
    c. Attachments. If attachments are not accepted, please upload them in a Google drive link and send the Share link to me instead.
10. All students who would not be able to comply with the given deadlines will be assessed on a case to case basis.
11. Rubrics will be uploaded soon.