

Question 1: (10 marks)

Write a C program, q1.c, to take a list of command line arguments, each of which is the full path of a command (such as /bin/ls, /bin/ps, /bin/date, /bin/who, /bin/uname etc). Assume the number of such commands is N. Your program would create N direct child processes. The parent of these child processes is your program. Each child executes one of the N commands. You should make sure that these N commands are executed concurrently, not sequentially one after the other. The parent process should be waiting for each child process to terminate. When a child process terminates, the parent process should print one line on the standard output stating that the relevant command has completed successfully or not successfully (such as "Command /bin/who has completed successfully", or "Command /bin/who has not completed successfully"). Once all the children processes have terminated, the parent process should print "All done, bye!" before it itself terminates. Note: Do not use function system in this question. You can use the different types of exec function.

Question 2: (10 marks)

Write a C program, q2.c, like the standard Unix utility ls -l but with much less functionality. Specifically, it takes a list of command line arguments, treating each command line argument as a file name. It then reports the following information for each file: User and Group names of the user and group owners, the type of file, full access permissions (reported in the format used by the ls program), the size of the file, i-node number, the device number of the device in which the file is stored, the number of links, last access, modification and file status changes time (converted to the format used by the ls program). Like ls -l command, if no command line argument is provided, the program simply reports the stated information about the files in the current directory. Your program cannot use /bin/ls program, /usr/bin/stat program, or any other program to implement the program for this question.

Question 3: (10 marks)

Write a C program, q3.c, to implement a simple shell accepting only 4 commands. The first is the copy command cp (i.e. cp file1 file2) to copy all the content from file1 to file2. The second is a command to show the current directory (pwd). The third command is the I/O redirection command (pwd > foo). The final is the exit command to quit the shell.

Question 4: (20 marks)

Semaphore is a type of inter-process communication mechanism used to facilitate process synchronisation. Write a C program, q4.c, to illustrate the 3 synchronisation problems (Deadlock, Starvation, and Race condition). You can put all in a single c file or 3 c files (e.g. q4deadlock.c, q4starvation.c, q4race.c). Your code needs to clearly demonstrate the respective problems. The quality of your solution will base on the complexity of the scenario. The scenario requires minimal 5 processes and 3 resources (files). Discuss 1 solution for each problem. Present the code of your solution with the individual problem if possible. Clearly document the problem scenarios and the solution with a detail discussion.

ICT1007 20/21 Sem 2
Assignment 2
(4/4/2021 23:59)

Document requirements for each question:

- Brief introduction of the objective of the code and a listing the functionalities.
- Design and analysis discussion.
- Results: Screen shots of how your program fulfils the functionalities. If your programme is not working, honestly declare it and explain what the possible problem can be.
- Testing: Other testing results to show that your code is robust.
- Source code listing.

Contribution statement:

| Name | Description of Task | Percentage | Signature | Date |
|-------|---------------------|------------|-----------|------|
| 1. | | | | |
| 2. | | | | |
| 3. | | | | |
| 4. | | | | |
| 5. | | | | |
| Total | | 100% | | |

Team effort discussion.

*Note: There should be consensus of the level of each member's contribution. The total contribution must add up to 100%. This statement must be signed by all group members and submitted with the report.

Marking Guide:

| | Marks | |
|------------------|-------|---|
| Question 1,2,3,4 | 50 | The code must be working according to what it has stated in the question. The quality will depend on the robustness, readability, and efficiency of your code. |
| Demonstration | 10 | Demonstration is required during the last tutorial. Quality will depend on the organisation and clarity of the demonstration. |
| Document | 30 | Intro, design and analysis (10), Result and testing (10), Source code listing with good programming practice and style and report structure and organisation (10) |
| Teamwork | 10 | Contribution statement and a discussion of your team effort. |

*Note: Marks may vary for individual member depending on the contribution and the performance during demonstration.