CS307 – System Practicum

Feb-June 2021

Assignment 1

Course Instructor: *Dr. Aditya Nigam*

Date: 24th February 2021

Instructions

- Plagiarism is strictly prohibited. In case of violation, a zero will be awarded for this assignment as a warning and a quick F grade if repeated later.
- In Moodle, students need to submit below given files in a *zip* archive named as *{Group_id}*_assignment_1.zip:
  - A **readme.md** file which provides the instructions for running your codes in detail including the versions of programming language and all the modules that have been used.
  - **Makefile** for compiling and executing the given code, make sure to *use gcc <= 9.3.*
  - The **code** and **report** for the assignment.
  - Give appropriate names to all the submitted files.
- Submit a single report for the assignment. The report should be concise.
- Students using Windows or any other OS are requested to make sure that their code runs perfectly on Linux as mentioned in each problem. Your evaluation will be done on computers having Ubuntu 20.04.
- Students must write their codes in C/C++ with gcc <= 9.3 programming language.
- The deadline for submission is **8th March 2021**. No late submissions will be entertained.
- Contact **Abhishek** [*b17072@students.iitmandi.ac.in*], **Daksh** [*dak.thapar@gmail.com*] or **Ranjeet** [*ranjanjharanjeet@gmail.com*] for any queries.

Problem 1 – Creating a Shell

(15 marks)

The shell or Command line interpreter is a fundamental User Interface in an Operating System. You have to create a simple shell that supports the following internal commands:

1. **clr**
   Clear the screen. (1 mark)

2. **pause**
   Pause operations of the shell until 'Enter' is pressed. (1 mark)

3. **help**
   Display User Manual. (1 mark)

4. **quit** / **Ctrl+D**
   Quit the shell. (1 mark)

5. **history**
   Display the list of previously executed commands, even on shell restart. (1 mark)

6. **cd <directory>**
   Change the current default directory to <directory>. If the <directory> argument is not present, report the current directory. If the directory doesn't exist an appropriate error should be reported. This command should change the PWD environment variable for current shell. (1 mark)

7. **dir <directory>**
   List all the contents of the directory <directory>. (1 mark)

8. **environ**
   List all the environment strings of the current shell and the bash shell. (1 mark)

9. **echo <comment>**
   Display <comment> on the display followed by a new line. Multiple spaces/tabs should be reduced to a single space. (1 mark)

10. The shell environment should contain **shell=<pathname>=myshell** where **<pathname>=myshell** is full path for the shell executable (not a hardwired path back to your directory, but the one from which it was executed). (3 marks)

11. The shell must be able to take its command line input from a file. That is, if the command line is invoked with a command line argument: **myshell <batchfile>** then <batchfile> is assumed to contain a set of command lines for the shell to process.

When the end of file is reached, the shell should exit. Obviously, if the shell is invoked without a command line argument, it solicits input from the user via prompt on the display. (3 marks)

Note:

1. Do not use wrapper function unless essential.
2. Avoid using direct system calls to implement commands unless essential.
3. Do appropriate error handling wherever necessary.

Problem 2 – Dining students

(15 marks)

5 students love the canteen food. They sit around a canteen table and eat food using a pair of spoons. There is a food bowl at the centre of the table. Each student has a left spoon and a right spoon (a total of 5 spoons on the table). The job of each student is to eat and think repeatedly. But one can eat only when both of their left and right spoons are available. The eat-think cycle of a student is as follows: pick up one spoon, pick up other spoon, eat food for 20 second, put down both the spoons, and think for a random amount of time (< 2 seconds).

Use threads with locks to simulate the students eat-think lifecycle. There are 4 states for each student:

1. Waiting for Spoons
2. One spoon acquired.
3. Both spoons acquired and Eating (20 seconds)
4. Thinking (< 2 seconds)

Output:

1. A file containing the state change log. We call it a change when the state (waiting for spoons/one spoon acquired/both spoons acquired and eating/thinking) of even one student changes.
   For example, consider an instant where student 1 (S1) has both spoons and eating, student 2 (S2) is waiting for spoons, student 3 (S3) has one spoon acquired, and all other students are thinking. The next instant, student 4 (S4) comes out of thinking. Rest remains same for this instant. The state change log will look like:

   S1: Both spoons acquired and eating.
   S2: Waiting for spoons
   S3: One spoon acquired
   S4: Thinking
   S5: Thinking

S1: Both spoons acquired and eating.
S2: Waiting for spoons
S3: One spoon acquired
S4: Waiting for spoons
S5: Thinking

2. Run your code 5 times for a simulation time of 30 minutes each. Generate the 5 different state change log files and summarize your findings in a report. Discuss whether your solution is mutually exclusive, prevents deadlock and starvation i.e. allows **exactly** fair allocation.

Problem 3 – Matrix Multiplication

(10 marks)

Write a sequential program to multiply two square matrices. Initialize the matrices with any random values. The size of matrix (n) should be taken as command line argument. Now write a parallel program using threads to do the same. Run both programs for different values of n (n=1 to at least 3000) and make proper log of respective running time.

Input

n = size of the square matrix

Output

Create a report which includes the graph of input size (n) vs running times for both programs. Run each program at least 5 times and use average running time for analysis. Discuss your inferences.

Note

1. Use dynamic memory allocation for arrays.
2. Avoid running any heavy program at the time of execution.

The End