New York University

Tandon School of Engineering

Department of Computer Science and
Engineering

Introduction to Operating Systems

Fall 2024

Assignment 4 (70 points)

Problem 1 (10 points) If you create a main() routine that calls fork() four times, i.e., if it includes the following code:

```
    pid_t w=-11, x=-22, y=-33, z=-44;
w = fork();
if(w==0) x = fork();
if(x==0) y = fork();
if(y>0) z = fork();
```

Assuming all fork() calls succeed, draw a process tree similar to that of Fig. 3.8 (page 116) in your textbook, clearly indicating the values of w, x, y, and z for each process in the tree (i.e., whether 0, -11, -22, -33, -44, or larger than 0). Note that the process tree should only have one node for each process and thus the number of nodes should be equal to the number of processes. The process tree should be a snapshot just after all forks completed but before any process exists. Each line/arrow in the process tree diagram shall represent a creation of a process, or alternatively a parent/child relationship.

Problem 2 (10 points) Write a program that creates the process tree shown below:

Problem 3 (10 points) Write a program whose main routine obtains a parameter n from the user (i.e., passed to your program when it was invoked from the shell, n¿2) and creates a child process. The child process shall then create and print a Fibonacci sequence of length n and whose elements are of type `unsigned long long`. You may find more information about Fibonacci numbers at (https://en.wikipedia.org/wiki/Fibonacci_number). The parent waits for the child to exit and then prints two additional Fibonacci elements, i.e., the

total number of Fibonacci elements printed by the child and the parent is n+2. Do not use IPC in your solution to this problem (i.e., neither shared memory nor message passing).

Problem 4 (10 points) Explain the differences between a process and a thread. Discuss the advantages and disadvantages of using threads versus processes. Provide examples of when you might choose one over the other.

Problem 5 (10 points) Write a C program that uses threads to compute the sum of the numbers from 1 to 1000000. Divide the work among multiple threads, and then combine their results to get the final sum. Print the final sum and the time it took to compute it.

Problem 6 (10 points) Describe the concept of a race condition. Explain how mutexes can be used to prevent race conditions. Illustrate your explanation with a simple C program that demonstrates a race condition and then show how to fix it using a mutex.

Problem 7 (10 points) Design a producer-consumer problem solution using semaphores in C. The producer generates random numbers and adds them to a bounded buffer. The consumer removes numbers from the buffer and prints them. Ensure that the producer doesn't add to a full buffer and the consumer doesn't remove from an empty buffer. Use appropriate semaphore operations to synchronize the producer and consumer.

What to hand in (using Brightspace):

Please submit the following files individually:

1) Source file(s) with appropriate comments. The naming should be similar to "lab#_$.c" (# is replaced with the assignment number and $ with the question number within the assignment, e.g., `lab4_b.c`, for lab 4, question c OR `lab5_1a` for lab 5, question 1a).

2) A single pdf file (for images + report/answers to short-answer questions), named "lab#.pdf" (# is replaced by the assignment number), containing:

- Screenshot(s) of your terminal window showing the current directory, the command used to compile your program, the command used to run your program, and the output of your program.

- Your Makefile, if any. This is applicable only to kernel modules.

RULES:

- You shall use kernel version 4.x.x or above. You shall not use kernel version 3.x.x.

- You may consult with other students about GENERAL concepts or methods but copying code (or code fragments) or algorithms is NOT ALLOWED and is considered cheating (whether copied from other students, the internet, or any other source).

- If you are having trouble, please ask your teaching assistant for help.

- You must submit your assignment prior to the deadline.