New York University
Tandon School of Engineering
Department of Computer Science and Engineering
Introduction to Operating Systems
Fall 2024
Assignment 4 (10 points)

Problem 1 (2 points)

If you create a main() routine that calls fork() three times, i.e., if it includes the following code:

```
   pid_t x=-11, y=-22, z=-33;
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 x, y and z for each process in the tree (i.e., whether 0, -11, -22, -33, 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.

(Insert process tree diagram here - This should show 8 processes. The root process will have x ¿ 0, y = -22, z = -33. Its children will have x = 0, y ¿ 0, z = -33 and x = 0, y = -22, z = -33 respectively. This pattern will continue)

Problem 2 (4 points)

Write a program that creates the process tree shown below:

(Insert process tree diagram here - This should be a tree with a root process, and that root process creating 2 children, and each of those children creating 1 child each. A total of 5 processes.)

Problem 3 (4 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 sequence of n numbers where each number is the sum of the previous two numbers. The first two numbers in the sequence are 1 and 1. The elements are of type `unsigned long long`. The parent waits for the child to exit and then prints an additional element following the sequence. The total number of elements printed by the child and the parent is n+1. Do not use IPC in your solution to this problem (i.e., neither shared memory nor message passing).

Problem 4 (5 points)

Consider a system with three processes, P1, P2, and P3. They share a single resource that can be used by only one process at a time. Each process needs to use the resource for a certain amount of time (burst time). Assume the following:

* P1: Burst time = 5 units * P2: Burst time = 3 units * P3: Burst time = 7 units

The processes arrive at the following times:

* P1: Time 0 * P2: Time 2 * P3: Time 4

Illustrate the execution using a Gantt chart for both First-Come, First-Served (FCFS) and Shortest Job First (SJF) scheduling algorithms. Calculate the average waiting time for each algorithm. Show your calculations.

Problem 5 (5 points)

Explain the concept of a deadlock in operating systems. Describe four necessary conditions for a deadlock to occur. Give a real-world example of a deadlock (not related to computers). Provide an example of a deadlock scenario involving three processes and two resources, using a resource allocation graph.

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 ques-

tion 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. (For Problems 2 and 3)
- Gantt charts for Problem 4.
- Resource allocation graph and explanation for Problem 5.

3) 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.