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() twice, i.e., if it includes the following code:

```
pid_t x=-11, y=-22;
x = fork();
if(x==0) y = 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 and y for each process in the tree (i.e., whether 0, -11, -22, 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 exits. Each line/arrow in the process tree diagram shall represent a creation of a process, or alternatively a parent/child relationship.

Process Tree Diagram (Insert Diagram Here - Should show 3 processes)

Problem 2 (4 points)

Write a program that creates the process tree shown below:

(Insert Process Tree Diagram Here - Should show a tree with at least 4 processes and clear parent-child relationships)

Problem 3 (4 points)

Write a program whose main routine obtains a parameter n from the user (passed to your program when invoked from the shell, n¿2) and creates a child process. The child process shall then calculate and print the sum of even numbers from 1 to n. The parent waits for the child to exit and then prints the sum of odd numbers from 1 to n. Do not use IPC in your solution to this problem (i.e., neither shared memory nor message passing).

Listing 1: Example C Code

```
//  Insert C code here
```

Problem 4 (2 points)

Explain the difference between a process and a thread. Give examples of when you might choose to use threads over processes, and vice-versa.

Problem 5 (2 points)

Describe the concept of a zombie process. How are zombie processes created, and how can they be avoided?

Problem 6 (4 points)

Write a C program that uses signals to handle a SIGINT (Ctrl+C) signal. When a SIGINT signal is received, the program should gracefully terminate and print a message indicating that it is shutting down.

Problem 7 (2 points)

Draw a state diagram illustrating the different states a process can be in during its lifecycle (e.g., New, Ready, Running, Blocked, Terminated). Include transitions between states and label the transitions with the events causing the transitions (e.g., scheduler dispatch, I/O request completion).

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 b 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.

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.