Tutorial 3, CSN-232

Q1. The Fibonacci sequence is the series of numbers 0, 1, 1, 2, 3, 5, 8, ... Formally, it can be expressed as:

$$\begin{aligned} fib_0 &= 0 \\ fib_1 &= 1 \\ fib_n &= fib_{n-1} + fib_{n-2} \end{aligned}$$

Write a C program using the fork() system call that that generates the Fibonacci sequence in the child process. The number of the sequence will be provided in the command line. For example, if 5 is provided, the first five numbers in the Fibonacci sequence will be output by the child process. Because the parent and child processes have their own copies of the data, it will be necessary for the child to output the sequence.

Have the parent invoke the wait() call to wait for the child process to complete before exiting the program. Perform necessary error checking to ensure that a non-negative number is passed on the command line.

Q2. Space sharing is an approach in multiprocessor scheduling environments when the processes are related to one another. Assuming that an entire group of related threads is created at once, describe the basic functioning of space sharing approach along with advantage and disadvantages.

Q3. Consider a multiprocessor system and a multithreaded program written using the many-to-many threading model. Let the number of user-level threads in the program be more than the number of processors in the system. Discuss the performance implications of the following scenarios.

a. The number of kernel threads allocated to the program is less than the number of processors.

b. The number of kernel threads allocated to the program is equal to the number of processors.

c. The number of kernel threads allocated to the program is greater than the number of processors but less than the number of user-level threads.

Q4. A process main() takes n integers as input and stores the integers in a comma separated file. Then it calls a child process, C1, with execv(), also passing the csv file path. Now C1 reads the csv file and sums all the integers and writes the sum back in a file named *sum* in the same directory. After the file is written, the main() process reads the sum file and displays the value on console.

Each process should show its process id at the start. Write a program to implement the above scenario.