Read Chapter 5 – CPU Scheduling

The CPU scheduler is an important component of the operating system. Processes must be properly scheduled, or else the system will make inefficient use of its resources. Different operating systems have different scheduling requirements, for example a supercomputer aims to finish as many jobs as it can in the minimum amount of time, but an interactive multi-user system such as a Windows terminal server aims to rapidly switch the CPU between each user in order to give users the “illusion” that they each have their own dedicated CPU.

Which is the best CPU scheduling algorithm? There is no hard-and-fast answer, but one way to find out is to simulate different scheduling algorithms with the type of jobs your system is going to be getting, and see which one is the best. This is what you will be doing for this assignment.

There are two parts to this assignment:

1. Implementation of a CPU scheduler simulation to compare two schedules described in Chapter 5 (use Java, C, or C++); and

2. Create a 1-2 page report describing your evaluation of these different scheduling algorithms. Did one work better then the other? Which algorithm might be better then another for a given purpose?

The Simulator

A job can be defined by an arrival time and a burst time. For example, here’s a sequence of jobs:

<0, 100>, <2, 55>, <2, 45>, <5, 10>…

The first job arrives at time 0 and requires 100ms of CPU time to complete; the second job arrives at time 2 and requires 55ms of CPU time; the third job arrives at time 2 and requires 45ms; and so on. You can assume that time is divided into millisecond units.

Your simulator should first generate a sequence of jobs. The burst lengths can be determined by selecting a random number from an exponential distribution.
There should also be a minimum job length of 2ms, so that the total burst duration for a job is 2ms plus the value selected from the exponential distribution (which should be between 0 and 40). So the shortest job will require for 2ms of CPU time and the longest, 42ms.

Your program should simulate the arrival of jobs for up to \( n \) milliseconds and then stop. Once the jobs have been generated, you will need to compare the performance of different scheduling algorithms on the same set of jobs. You can write one program that runs both algorithms or write two separate programs.

For each scheduling algorithm, your program should measure at least (1) the CPU utilization, (2) the average job throughput per second, and (3) the average job turnaround time. These statistics are described on pg. 187 of the textbook.