Joseph Moran

24 October 2025

CS 4310 – Operating Systems

Project 1 Report

This project implements two main components: a multithreaded sorting program and several CPU scheduling algorithms, including FCFS, SJF, preemptive priority, and round-robin. The focus of this report is to explain the structure of the code and how loops and conditional statements were used to control program flow and make decisions.

The sorting program divides an array into two halves and uses two separate threads to perform bubble sort concurrently on each half. A third thread merges the two sorted halves into a single array. For loops are used extensively in the bubble sort function to iterate over array elements and swap values when necessary. Conditional statements check whether one element is greater than another before performing swaps. In the merge function, while loops are used to compare elements from the two halves and append the smaller element to the merged array. Additional while loops handle any remaining elements once one half is exhausted. These loops and conditionals ensure that the sorting process is efficient and correct.

The scheduling algorithms work with tasks read from an input file. Each task has a process ID, arrival time, burst time, and priority. Loops are used to iterate through the list of processes, check which tasks are ready to run, and update their remaining times. Conditional statements determine whether a process has arrived, whether it has completed, and which process to select next based on the scheduling algorithm. For example, in the round-robin algorithm, a while loop continues until all processes are completed. Inside this loop, if statements check for new arrivals and decide whether to run a process for a full time quantum or until completion. In preemptive priority scheduling, if statements compare priorities to select the highest-priority ready process.

In conclusion, the combination of for loops, while loops, and conditional statements was central to the design and functionality of both the multithreaded sorting program and the CPU scheduling algorithms. These constructs allowed the program to systematically process arrays and tasks, make decisions based on dynamic conditions, and maintain correct program behavior while handling concurrency and process management.

Output:

```
PS C:\Users\jmora\4310-Projects\Project1> .\build\Debug\Project1.exe
Sorted array:
1 2 3 4 5 7 8 10 15 20 40
Choose a scheduling algorithm:
1 - FCFS
2 - SJF
3 - Preemptive Priority
4 - Round Robin
Enter choice: 3
Time 1 ms: Process 1 is running.
Time 2 ms: Process 1 is running.
Time 3 ms: Process 2 is running.
Time 4 ms: Process 2 is running.
Time 5 ms: Process 2 is running.
Time 6 ms: Process 1 is running.
Time 7 ms: Process 1 is running.
Time 8 ms: Process 1 is running.
Time 9 ms: Process 4 is running.
Time 10 ms: Process 4 is running.
Time 11 ms: Process 6 is running.
Time 12 ms: Process 6 is running.
Time 13 ms: Process 6 is running.
Time 14 ms: Process 6 is running.
Time 15 ms: Process 4 is running.
Time 16 ms: Process 4 is running.
Time 17 ms: Process 4 is running.
Time 18 ms: Process 4 is running.
Time 19 ms: Process 10 is running.
Time 20 ms: Process 10 is running.
Time 21 ms: Process 8 is running.
Time 22 ms: Process 8 is running.
Time 23 ms: Process 8 is running.
Time 24 ms: Process 3 is running.
Time 25 ms: Process 3 is running.
Time 26 ms: Process 3 is running.
Time 27 ms: Process 3 is running.
Time 28 ms: Process 3 is running.
Time 29 ms: Process 3 is running.
Time 30 ms: Process 3 is running.
Time 31 ms: Process 3 is running.
Time 32 ms: Process 7 is running.
Time 33 ms: Process 7 is running.
Time 34 ms: Process 7 is running.
Time 35 ms: Process 7 is running.
Time 36 ms: Process 7 is running.
Time 37 ms: Process 7 is running.
Time 38 ms: Process 7 is running.
Time 39 ms: Process 5 is running.
Time 40 ms: Process 5 is running.
Time 41 ms: Process 9 is running.
Time 42 ms: Process 9 is running.
Time 43 ms: Process 9 is running.
Time 44 ms: Process 9 is running.
Time 45 ms: Process 9 is running.
Preemptive Priority Results:
Average Waiting Time: 10.7
Average Turnaround Time: 15.2
CPU Utilization: 100%
```