

# ***CSC227***

## ***Operating systems***

### ***Programming Assignment1 Process Scheduling Simulator***

Prepared by

Group members:
<i>Rand Al-bargan, 443200839, Leader</i>
<i>Joud Bakarman, 443201057, Member</i>
<i>Sara Aljabali, 444201613, Member</i>
<i>Aljawharah Alsaab, 444200860, Member</i>
<i>Bayan Alghamdi, 443204310, Member</i>

## *Work Distribution*

<i>task</i>	<i>member</i>
<i>Implement Scheduling Algorithm</i>	Rand Al-barqan & Bayan Alghamdi
<i>Simulation &amp; Time Handling</i>	Sara Aljabali & Joud Bakarman
<i>Performance Metrics Calculation</i>	Aljawharah Alsaab
<i>Visualization</i>	Rand Al-barqan
<i>Documentation</i>	Sara Aljabali & Joud Bakarman

### Test Case 1 (Example output):

```

SRTF - Apache NetBeans IDE 12.2
Output - SRTF (run) x
Enter number of processes: 4
Enter Arrival Time for P1: 0
Enter Burst Time for P1: 8
Enter Arrival Time for P2: 1
Enter Burst Time for P2: 4
Enter Arrival Time for P3: 2
Enter Burst Time for P3: 5
Enter Arrival Time for P4: 3
Enter Burst Time for P4: 5

Scheduling Algorithm: Shortest Remaining Time First
Context Switch: 1 ms
Time    Process/CS
0-1     P1
1-2     CS
2-6     P2
6-7     CS
7-12    P3
12-13   CS
13-18   P4
18-19   CS
19-26   P1

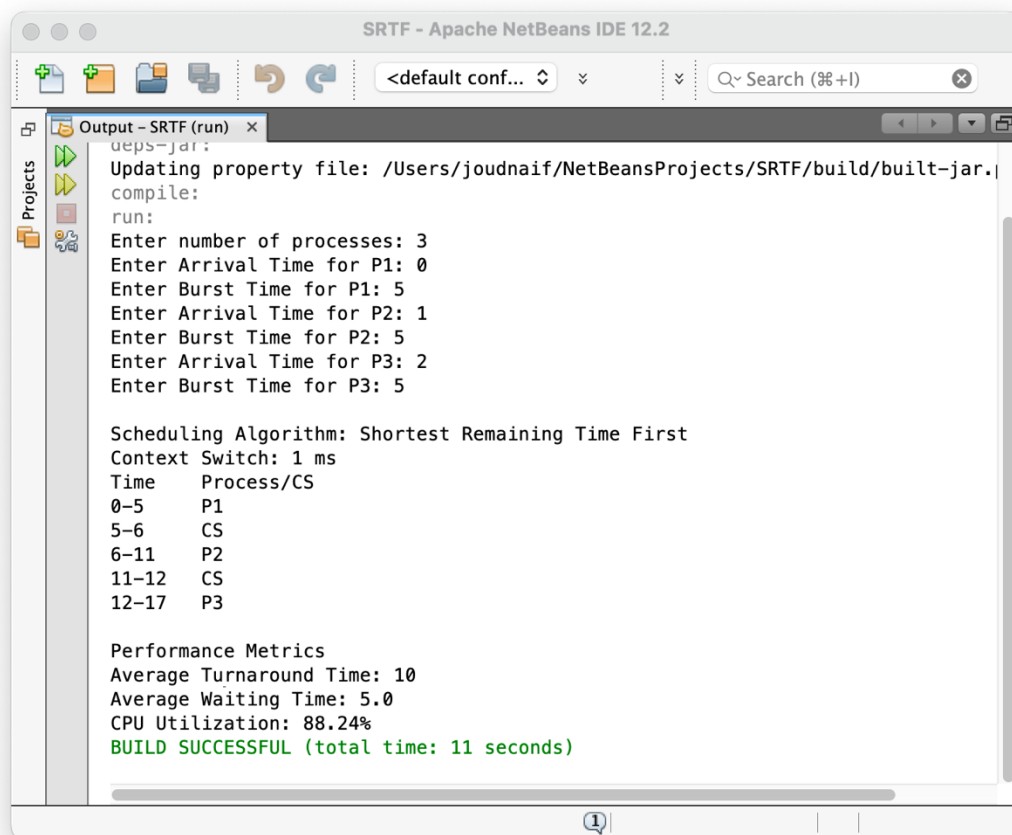
Performance Metrics
Average Turnaround Time: 14
Average Waiting Time: 8.5
CPU Utilization: 84.62%
BUILD SUCCESSFUL (total time: 31 seconds)
  
```

This test case evaluates the Shortest Remaining Time First (SRTF) scheduling algorithm with four processes arriving at different times. The scheduler selects the process with the shortest remaining burst time for execution. A context switch (CS) of 1 ms is introduced whenever a process change occurs.

*Execution Breakdown:*

1. **P1** starts execution at time 0.
2. **P2** arrives at time 1 and preempts **P1** after a context switch.
3. **P2** executes from time 2 to 6 and completes.
4. **P3** (shorter remaining time) starts execution after a CS, running from 7 to 12 before being preempted.
5. **P4** begins execution at time 13 and runs until time 18 before being preempted.
6. **P1** resumes execution at time 19 and completes at time 26.

## Test case 2:



```
Output - SRTF (run) x
ueps-jd1:
Updating property file: /Users/joudnaif/NetBeansProjects/SRTF/build/built-jar.
compile:
run:
Enter number of processes: 3
Enter Arrival Time for P1: 0
Enter Burst Time for P1: 5
Enter Arrival Time for P2: 1
Enter Burst Time for P2: 5
Enter Arrival Time for P3: 2
Enter Burst Time for P3: 5

Scheduling Algorithm: Shortest Remaining Time First
Context Switch: 1 ms
Time    Process/CS
0-5     P1
5-6     CS
6-11    P2
11-12   CS
12-17   P3

Performance Metrics
Average Turnaround Time: 10
Average Waiting Time: 5.0
CPU Utilization: 88.24%
BUILD SUCCESSFUL (total time: 11 seconds)
```

*This test case works with three processes, all having equal burst times. Since SRTF prioritizes the process with the shortest remaining burst time, when multiple processes have the same burst time, the algorithm follows First-Come, First-Served (FCFS) order to break the tie. This test case effectively demonstrates how SRTF behaves when multiple processes have the same next CPU burst, leading to FCFS execution among them.*

*Execution Breakdown:*

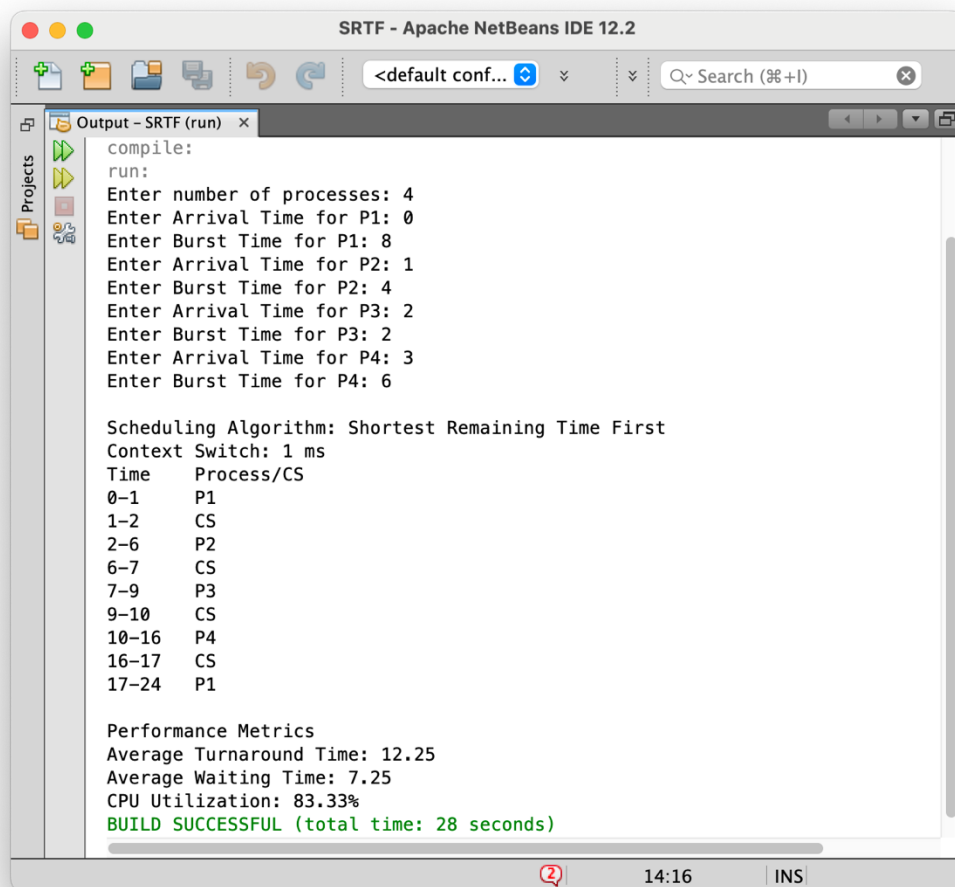
1. **P1** starts execution at time 0 and runs until time 5.
  - Since no other process with a shorter burst time has arrived, **P1** executes without preemption.

2. At time 5, a context switch occurs, and **P2** begins execution from time 6 to 11.
  - **P2** and **P3** have the same burst time (5), but **P2** arrived earlier (at time 1), so it is scheduled first following FCFS.
3. At time 11, a context switch occurs, and **P3** executes from time 12 to 17.
  - Since **P3** was the last process in the queue and has no competition, it executes without interruption.

*How This Test Case Demonstrates the FCFS Condition:*

1. **P1** executes first because it arrives earliest, even though all processes have the same burst time.
2. **P2** and **P3** both have equal burst times (5), but **P2** is selected first because it arrives earlier (FCFS rule).
3. SRTF does not preempt any process in this case since all have identical CPU bursts, effectively making it an FCFS scenario.

### Test Case 3:



```

SRTF - Apache NetBeans IDE 12.2
Output - SRTF (run) x
compile:
run:
Enter number of processes: 4
Enter Arrival Time for P1: 0
Enter Burst Time for P1: 8
Enter Arrival Time for P2: 1
Enter Burst Time for P2: 4
Enter Arrival Time for P3: 2
Enter Burst Time for P3: 2
Enter Arrival Time for P4: 3
Enter Burst Time for P4: 6

Scheduling Algorithm: Shortest Remaining Time First
Context Switch: 1 ms
Time    Process/CS
0-1     P1
1-2     CS
2-6     P2
6-7     CS
7-9     P3
9-10    CS
10-16   P4
16-17   CS
17-24   P1

Performance Metrics
Average Turnaround Time: 12.25
Average Waiting Time: 7.25
CPU Utilization: 83.33%
BUILD SUCCESSFUL (total time: 28 seconds)
  
```

Lastly, this test case effectively showcases preemption, where a newly arriving process with a shorter burst time preempts the currently executing process.

*Execution Breakdown with Preemption and FCFS Condition:*

1. **P1** starts execution at time 0 but is preempted by **P2** at time 1
  - Preemption: **P2** has a shorter burst time (4) than **P1**'s remaining burst time (7).
  - A context switch occurs at time 1.
2. **P2** starts execution at time 2 but is preempted by **P3** at time 6
  - Preemption: **P3** has a shorter burst time (2) than **P2**'s remaining burst time (4).
  - A context switch occurs at time 6.
3. **P3** executes from time 7 to 9 and completes execution.
  - A context switch occurs at time 9.
4. **P4** starts execution at time 10 and runs until time 16.
  - No preemption occurs because **P4** has the shortest remaining burst time.
  - A context switch occurs at time 16.
5. **P1** resumes execution at time 17 and completes at time 24.
  - Since all other processes have completed, **P1** runs until completion.

---

*How This Test Case Demonstrates Preemption:*

1. **P2** preempts **P1** at time 1 because **P2** has a shorter burst time (4) than **P1**'s remaining time (7).
2. **P3** preempts **P2** at time 6 because **P3** has a burst time (2) shorter than **P2**'s remaining time (4).
3. **P1** is postponed multiple times due to preemptions but eventually completes last.