Operating Systems Workbench V2.1
Scheduling (Advanced) Activities and Experiments

Richard Anthony   January 2005

This laboratory sheet accompanies the '***Scheduling Algorithms – Advanced***' application within the Operating Systems Workbench.

1. Prerequisite knowledge

You should have a clear understanding of the following concepts: Operating System, Processor, Process.
You should have a basic understanding of the following concepts: Blocking, Preemption.
You should be familiar with the '***Scheduling Algorithms – Introductory***' application within the Operating Systems Workbench.

2. Introduction

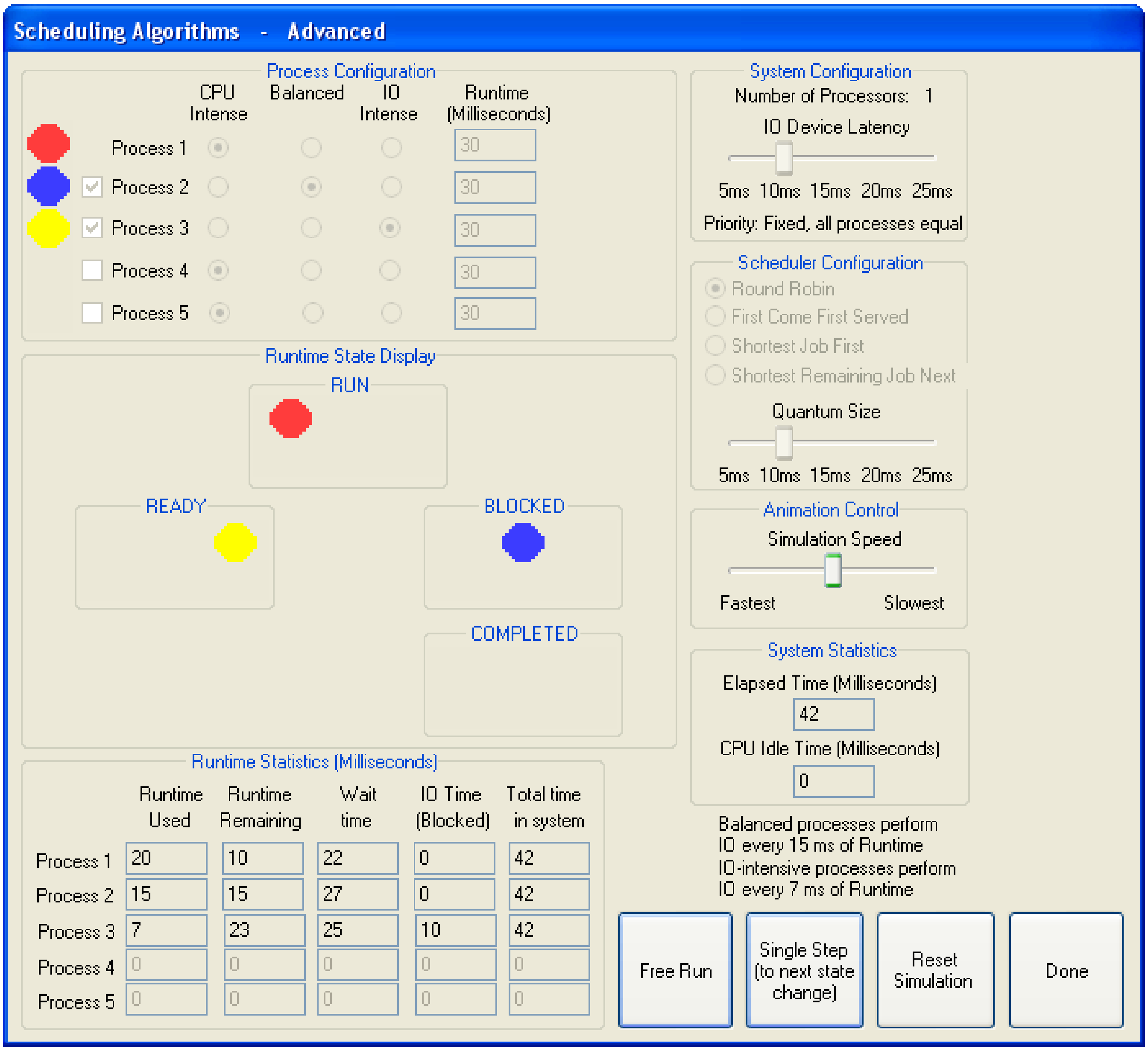
This simulation has been designed to introduce you to the fundamentals of scheduling whilst keeping the complexity low. The simulation shows scheduling behaviour in terms of the movement of processes between the RUN, READY and BLOCKED states.
You can investigate the way in which four different scheduling algorithms work with between one and five processes. The processes can be configured to be one of: CPU intensive (does no Input / Output); Balanced (does some Input / Output); or IO intensive (does lots of Input / Output) – you will be able to investigate how the tasks' characteristics affect their behaviour in the system.

Figure 1 shows the Scheduling (advanced) interface during a simulation of Round Robin Scheduling with three processes.

The display is divided into a number of sections. Each section is briefly explained:

- **Process Configuration** – this enables configuration of the number of processes and the type of processes used in the simulation. In the example shown three processes are enabled. Process 1 is configured to be CPU intensive (that is, it does no I/O) Process 2 is configured to be Balanced (does some I/O) whilst Process 3 is configured to be I/O intensive (that is, it does lots of I/O). All processes have been configured to have a runtime of 30 milliseconds (this is, the amount of CPU processing time required is 30 milliseconds, the actual time they will spend in the system can be more than this).
- **System Configuration** – This tells you the configuration settings for the scheduler itself. The IO device latency controls the amount of time a process will spend in the Blocked state each time it performs an IO operation (i.e. it is a measure of the operating speed of the IO device). In this example it has been set to 10 milliseconds.
- **Scheduler Configuration** – this enables you to select one of four different scheduling algorithms. It also enables you to configure the scheduling quantum size (the amount of CPU time given to a process at a time when preemptive scheduling is used). In this example it has been set to 10 milliseconds.
- **Animation Control** – this enables you to select the speed of the simulation.
- **System Statistics** – this provides a real-time display of the elapsed time since the start of a simulation. The amount of CPU idle-time is also displayed (this is the amount of time for which the CPU does not have any work to do).
- **Runtime Statistics** – this provides a real-time display of various statistics for each process during the simulation.
- **Runtime State Display** – This shows real-time scheduling behaviour in terms of the movement of the processes between the RUN, READY and BLOCKED states. A process enters the COMPLETED state when it has satisfied its processing requirement (as defined by its runtime value).

- **Free Run button** – this causes the simulation to run at the selected speed until all processes have completed, or the simulation is paused by pressing the Pause / Single-Step button.
- **Single-Step button** – this causes the simulation to run until the next process state-change and then stop. During free-run, this button's label changes to 'pause' and if pressed will stop the simulation at the next process state-change.
- **Reset Simulation button** – this button can be pressed at any time during a simulation. The process states and statistics are all reset but the simulation configuration settings are preserved. This is to permit unlimited repeat runs of the same simulation, without having to wait for completion.
- **Done button** – this button exits the application immediately without preserving statistics or configuration settings. Control is returned to the top-level menu.



Scheduling Algorithms - Advanced

Process Configuration

	CPU Intense	Balanced	IO Intense	Runtime (Milliseconds)
Process 1	●	○	○	30
✓ Process 2	○	●	○	30
✓ Process 3	○	○	●	30
☐ Process 4	●	○	○	30
☐ Process 5	●	○	○	30

System Configuration

Number of Processors: 1

IO Device Latency

5ms 10ms 15ms 20ms 25ms

Priority: Fixed, all processes equal

Scheduler Configuration

[✓ Round Robin]
First Come First Served
Shortest Job First
Shortest Remaining Job Next

Quantum Size

5ms 10ms 15ms 20ms 25ms

Animation Control

Simulation Speed

Fastest Slowest

System Statistics

Elapsed Time (Milliseconds): 42

CPU Idle Time (Milliseconds): 0

Balanced processes perform IO every 15 ms of Runtime
IO-intensive processes perform IO every 7 ms of Runtime

Runtime State Display

RUN ●

READY ●

BLOCKED ●

COMPLETED

Runtime Statistics (Milliseconds)

	Runtime Used	Runtime Remaining	Wait time	IO Time (Blocked)	Total time in system
Process 1	20	10	22	0	42
Process 2	15	15	27	0	42
Process 3	7	23	25	10	42
Process 4	0	0	0	0	0
Process 5	0	0	0	0	0

Figure 1. The Scheduling (advanced) interface.

The following 'lab activities' sections describe specific experiments and investigations to help you maximise the benefit of the software.

Please take care to read the instructions carefully for each step and try to follow instruction sequences carefully. Try to predict the outcome of experiments in advance if possible. If the outcome is not as expected try to determine why not. You can repeat experiments as often as required and can work at your own pace. Try repeating experiments with slight changes in the parameters – sometimes just changing one parameter slightly can lead to big differences in the results and can shed light on the relative importance of a particular aspect of the simulation.

For each activity the configuration settings are explained. In each case it is assumed that you have already started the ‘Advanced Scheduling Algorithms’ simulation application. To do this:
1. Start the **Operating Systems Workbench**.
2. From the main menu bar, select **Scheduling Algorithms**.
3. From the drop-down menu, select **Advanced**.

Lab Activity: Scheduling: Advanced: FCFS 1
Introduction to the FIRST COME FIRST SERVED (FCFS) Scheduling Algorithm

1. Configure the simulation as follows:
   - **Process 1**: Type = CPU Intense, Runtime = 30 milliseconds
   - **Process 2**: Type = CPU Intense, Runtime = 40 milliseconds
   - **Process 3**: Type = CPU Intense, Runtime = 50 milliseconds
   - **Process 4**: Type = CPU Intense, Runtime = 40 milliseconds
   - **Process 5**: Type = CPU Intense, Runtime = 30 milliseconds

   IO Device Latency = 10 milliseconds
   Quantum Size = 10 milliseconds
   Scheduler Configuration = First Come First Served

2. Press the **Start Configuration** button and pay attention to the sequence of states through which the processes pass.

3. When the process has completed, note the **System Statistics** and **Runtime Statistics**.

Questions

Q1. What is the total execution time of the processes (how long between starting and finishing)?

Q2. Do you think the FCFS scheduling algorithm is fair? Does fairness actually matter if all processes run for a very short time only? Try setting the process 1 Runtime to 300 milliseconds and repeat the experiment. Look at the wait time for each process – does this seem reasonable?

4. Change the configuration as follows:
   - **Process 1**: Type = Balanced, Runtime = 30 milliseconds
   - **Process 2**: Type = Balanced, Runtime = 40 milliseconds
   - **Process 3**: Type = Balanced, Runtime = 50 milliseconds
   - **Process 4**: Type = Balanced, Runtime = 40 milliseconds
   - **Process 5**: Type = Balanced, Runtime = 30 milliseconds

Q3. Do you think the FCFS scheduling algorithm is efficient (how often is the CPU idle).

Q4. From your observations, do you think the FCFS algorithm is preemptive or non-preemptive?

Lab Activity: Scheduling: Advanced: SJF 1
Introduction to the SHORTEST JOB FIRST (SJF) Scheduling Algorithm

1. Configure the simulation as follows:

   Process 1: Type = CPU Intense, Runtime = 30 milliseconds
   Process 2: Type = CPU Intense, Runtime = 40 milliseconds
   Process 3: Type = CPU Intense, Runtime = 50 milliseconds
   Process 4: Type = CPU Intense, Runtime = 40 milliseconds
   Process 5: Type = CPU Intense, Runtime = 30 milliseconds

   IO Device Latency = 10 milliseconds
   Quantum Size = 10 milliseconds
   Scheduler Configuration = Shortest Job First

2. Press the **Start Configuration** button and pay attention to the sequence of states through which the processes pass.

3. When the process has completed, note the **System Statistics** and **Runtime Statistics**.

Questions
Q1. What is the total execution time of the processes (how long between starting and finishing)?

Q2. Do you think the SJF scheduling algorithm is fair? Does fairness actually matter if all processes run for a very short time only? Try setting the process 1 Runtime to 300 milliseconds and repeat the experiment. Look at the wait time for each process – does this seem reasonable?

4. Change the configuration as follows:

   Process 1: Type = Balanced, Runtime = 30 milliseconds
   Process 2: Type = Balanced, Runtime = 40 milliseconds
   Process 3: Type = Balanced, Runtime = 50 milliseconds
   Process 4: Type = Balanced, Runtime = 40 milliseconds
   Process 5: Type = Balanced, Runtime = 30 milliseconds

Q3. Do you think the SJF scheduling algorithm is efficient (how often is the CPU idle).

Q4. From your observations, do you think the SJF algorithm is preemptive or non-preemptive?

Q5. What differences can you identify between the behaviour of SJF and that of FCFS?

Lab Activity: Scheduling: Advanced: RR 1
Introduction to the ROUND ROBIN (RR) Scheduling Algorithm

1. Configure the simulation as follows:
    Process 1: Type = CPU Intense, Runtime = 30 milliseconds
    Process 2: Type = CPU Intense, Runtime = 40 milliseconds
    Process 3: Type = CPU Intense, Runtime = 50 milliseconds
    Process 4: Type = CPU Intense, Runtime = 40 milliseconds
    Process 5: Type = CPU Intense, Runtime = 30 milliseconds

    IO Device Latency = 10 milliseconds
    Quantum Size = 10 milliseconds
    Scheduler Configuration = Round Robin

2. Press the **Start Configuration** button and pay attention to the sequence of states through which the processes pass.

3. When the process has completed, note the **System Statistics** and **Runtime Statistics**.

Questions
Q1. What is the total execution time of the processes (how long between starting and finishing)?

Q2. Do you think the RR scheduling algorithm is fair? Does fairness actually matter if all processes run for a very short time only? Try setting the process 1 Runtime to 300 milliseconds and repeat the experiment. Look at the wait time for each process – does this seem reasonable?

4. Change the configuration as follows:
    Process 1: Type = Balanced, Runtime = 30 milliseconds
    Process 2: Type = Balanced, Runtime = 40 milliseconds
    Process 3: Type = Balanced, Runtime = 50 milliseconds
    Process 4: Type = Balanced, Runtime = 40 milliseconds
    Process 5: Type = Balanced, Runtime = 30 milliseconds

Q3. Do you think the RR scheduling algorithm is efficient (how often is the CPU idle).

Q4. From your observations, do you think the RR algorithm is preemptive or non-preemptive?

Q5. What differences can you identify between the behaviour of RR and that of FCFS?

Q6. What differences can you identify between the behaviour of RR and that of SJF?

Lab Activity: Scheduling: Advanced: SRJN 1
Introduction to the SHORTEST REMAINING JOB NEXT (SRJN) Scheduling Algorithm

1. Configure the simulation as follows:
   Process 1: Type = CPU Intense, Runtime = 30 milliseconds
   Process 2: Type = CPU Intense, Runtime = 40 milliseconds
   Process 3: Type = CPU Intense, Runtime = 50 milliseconds
   Process 4: Type = CPU Intense, Runtime = 40 milliseconds
   Process 5: Type = CPU Intense, Runtime = 30 milliseconds

   IO Device Latency = 10 milliseconds
   Quantum Size = 10 milliseconds
   Scheduler Configuration = Shortest Remaining Job Next

2. Press the **Start Configuration** button and pay attention to the sequence of states through which the processes pass.

3. When the process has completed, note the **System Statistics** and **Runtime Statistics**.

Questions
Q1. What is the total execution time of the processes (how long between starting and finishing)?

Q2. Do you think the SRJN scheduling algorithm is fair? Does fairness actually matter if all processes run for a very short time only? Try setting the process 1 Runtime to 300 milliseconds and repeat the experiment. Look at the wait time for each process – does this seem reasonable?

4. Change the configuration as follows:
   Process 1: Type = Balanced, Runtime = 30 milliseconds
   Process 2: Type = Balanced, Runtime = 40 milliseconds
   Process 3: Type = Balanced, Runtime = 50 milliseconds
   Process 4: Type = Balanced, Runtime = 40 milliseconds
   Process 5: Type = Balanced, Runtime = 30 milliseconds

Q3. Do you think the SRJN scheduling algorithm is efficient (how often is the CPU idle).

Q4. From your observations, do you think the SRJN algorithm is preemptive or non-preemptive?

Q5. What differences can you identify between the behaviour of SRJN and that of FCFS?

Q6. What differences can you identify between the behaviour of SRJN and that of SJF?

Q7. What differences can you identify between the behaviour of SRJN and that of RR?

Lab Activity: Scheduling: Advanced: SRJN 2
Comparison of the SRJN and RR Scheduling Algorithms

Both SRJN and RR are preemptive. The objective of this activity is to compare the efficiency and fairness of the two algorithms.

1. Configure the simulation as follows:

   Process 1: Type = CPU Intense, Runtime = 40 milliseconds
   Process 2: Type = CPU Intense, Runtime = 40 milliseconds
   Process 3: Type = CPU Intense, Runtime = 40 milliseconds
   Process 4: Type = IO Intense, Runtime = 30 milliseconds
   Process 5: Type = IO Intense, Runtime = 30 milliseconds

   IO Device Latency = 10 milliseconds
   Quantum Size = 10 milliseconds
   Scheduler Configuration = Round Robin

2. Run the simulation and KEEP A RECORD OF ALL THE STATISTICS VALUES.

3. Modify the configuration as follows:

   Scheduler Configuration = Shortest Remaining Job Next

4. Run the simulation and KEEP A RECORD OF ALL THE STATISTICS VALUES.

Less CPU idle-time implies better scheduling efficiency.

Q1. Which algorithm was more efficient in these simulations? Is this difference due to a carefully chosen set of processes, or will it apply generally?

A 'fair' scheduler will give roughly similar waiting times to all processes.

Q2. Which algorithm was fairest in these simulations? Is this difference due to a carefully chosen set of processes, or will it apply generally?

Low Waiting-Time, and Low 'Total Time in the System' are measures of the responsiveness of processes.

5. Calculate the mean waiting-time, and mean total-time-in-system for the set of processes, for each simulation.

Q3. Which algorithm gave rise to the highest responsiveness of the processes in these simulations? Is this difference due to a carefully chosen set of processes, or will it apply generally?

Lab Activity: Scheduling: Advanced: Quantum Size 1
Investigation of the effects of changing the size of the Scheduling Quantum

1. Configure the simulation as follows:
   Process 1: Type = CPU Intense, Runtime = 40 milliseconds
   Process 2: Type = Balanced, Runtime = 50 milliseconds
   Process 3: Type = IO Intense, Runtime = 40 milliseconds
   Process 4: Type = Balanced, Runtime = 30 milliseconds
   Process 5: Type = IO Intense, Runtime = 30 milliseconds

   IO Device Latency = 10 milliseconds
   Scheduler Configuration = Round Robin
   Quantum Size = 5 milliseconds

2. Run the simulation and KEEP A RECORD OF ALL THE PROCESS STATISTICS VALUES.

3. Modify the configuration as follows:
   Quantum Size = 10 milliseconds

4. Run the simulation and KEEP A RECORD OF ALL THE PROCESS STATISTICS VALUES.

5. Modify the configuration as follows:
   Quantum Size = 15 milliseconds

6. Run the simulation and KEEP A RECORD OF ALL THE PROCESS STATISTICS VALUES.

7. Modify the configuration as follows:
   Quantum Size = 20 milliseconds

8. Run the simulation and KEEP A RECORD OF ALL THE PROCESS STATISTICS VALUES.

9. Modify the configuration as follows:
   Quantum Size = 25 milliseconds

10. Run the simulation and KEEP A RECORD OF ALL THE PROCESS STATISTICS VALUES.

Q1. How does the quantum size affect fairness (i.e. which quantum size causes the processes to encounter similar wait times, and which quantum size causes the greatest difference in the wait times)? Why is this so?

11. For each set of results, calculate the mean wait-time, and the mean total-time-in-the-system for the processes.

Q2. How does the quantum size affect the mean wait-time? Why is this?

Q3. How does the quantum size affect the mean total-time-in-the-system? Why is this?

Lab Activity: Scheduling: Advanced: IO Latency 1
Investigation of the effects of changing the IO device latency

1. Configure the simulation as follows:

    Process 1: Type = CPU Intense, Runtime = 40 milliseconds
    Process 2: Type = Balanced, Runtime = 50 milliseconds
    Process 3: Type = IO Intense, Runtime = 40 milliseconds
    Process 4: Type = Balanced, Runtime = 30 milliseconds
    Process 5: Type = IO Intense, Runtime = 30 milliseconds

    Quantum Size = 15 milliseconds
    Scheduler Configuration = Round Robin
    IO Device Latency = 5 milliseconds

2. Run the simulation and KEEP A RECORD OF ALL THE PROCESS STATISTICS VALUES.

3. Modify the configuration as follows:
    IO Device Latency = 10 milliseconds

4. Run the simulation and KEEP A RECORD OF ALL THE PROCESS STATISTICS VALUES.

5. Modify the configuration as follows:
    IO Device Latency = 15 milliseconds

6. Run the simulation and KEEP A RECORD OF ALL THE PROCESS STATISTICS VALUES.

7. Modify the configuration as follows:
    IO Device Latency = 20 milliseconds

8. Run the simulation and KEEP A RECORD OF ALL THE PROCESS STATISTICS VALUES.

9. Modify the configuration as follows:
    IO Device Latency = 25 milliseconds

10. Run the simulation and KEEP A RECORD OF ALL THE PROCESS STATISTICS VALUES.

Q1. As IO device latency increases what happens to the total-time-in-the-system (the time it takes to execute to completion) of each process?
Do any of the processes take longer to execute? If so, why?
Do any of the processes take less time to execute? If so, why?

11. For each set of results, calculate the mean wait-time, and the mean total-time-in-the-system for the processes.

Q2. How does the IO device latency affect the mean total-time-in-the-system? Why is this? Is this what you expected?

Q3. How does the IO device latency affect the mean wait-time? Why is this (think carefully)? Is this what you expected?

Lab Activity: Scheduling: Advanced: Predicting Behaviour 1
Predict and Analyse the behaviour of the scheduling algorithms under various circumstances

1. Configure the simulation as follows:

   Process 1: Type = CPU Intense, Runtime = 40 milliseconds
   Process 2: Type = Balanced, Runtime = 50 milliseconds
   Process 3: Type = IO Intense, Runtime = 30 milliseconds
   Process 4: Type = Not selected
   Process 5: Type = Not selected

   IO Device Latency = 10 milliseconds
   Quantum Size = 10 milliseconds
   Scheduler Configuration = Shortest Job First

Q1. Predict which process will finish first?
Q2. Predict which process will finish last?
Q3. Predict how much time process 2 will spend in the Blocked State?

2. Run the simulation to check your predictions. If you were wrong make sure that you understand why?

3. Modify the configuration as follows:

   Scheduler Configuration = Shortest Remaining Job Next

Q4. Predict which process will finish first?
Q5. Predict which process will finish last?
Q6. Predict how much time process 2 will spend in the Blocked State?

4. Run the simulation to check your predictions. If you were wrong make sure that you understand why?

Lab Activity: Scheduling: Advanced: Predicting Behaviour 2
Predict and Analyse the behaviour of the scheduling algorithms under various circumstances

1. Configure the simulation as follows:
   - **Process 1**: Type = Balanced, Runtime = 40 milliseconds
   - **Process 2**: Type = Balanced, Runtime = 50 milliseconds
   - **Process 3**: Type = IO Intense, Runtime = 30 milliseconds
   - **Process 4**: Type = Not selected
   - **Process 5**: Type = Not selected

   IO Device Latency = 20 milliseconds
   Quantum Size = 10 milliseconds
   Scheduler Configuration = Shortest Remaining Job Next

Q1. Predict which process will finish first?
Q2. Predict which process will finish last?
Q3. Predict how much time process 1 will spend in the Blocked State?

2. Run the simulation to check your predictions. If you were wrong make sure that you understand why?

3. Modify the configuration as follows:
   - **Process 1**: Type = CPU Intense, Runtime = 70 milliseconds
   - **Process 2**: Type = Balanced, Runtime = 50 milliseconds
   - **Process 3**: Type = IO Intense, Runtime = 30 milliseconds
   - **Process 4**: Type = Not selected
   - **Process 5**: Type = Not selected

Q4. Predict which process will finish first?
Q5. Predict which process will finish last?
Q6. Predict how much time process 1 will spend in the Blocked State?

4. Run the simulation to check your predictions. If you were wrong make sure that you understand why?

Lab Activity: Scheduling: Advanced: Predicting Behaviour 3
Predict and Analyse the behaviour of the scheduling algorithms under various circumstances

1. Configure the simulation as follows:

    Process 1: Type = IO Intense, Runtime = 40 milliseconds
    Process 2: Type = IO Intense, Runtime = 50 milliseconds
    Process 3: Type = CPU Intense, Runtime = 100 milliseconds
    Process 4: Type = IO Intense, Runtime = 30 milliseconds
    Process 5: Type = IO Intense, Runtime = 40 milliseconds

    Quantum Size = 25 milliseconds
    Scheduler Configuration = Shortest Remaining Job Next
    IO Device Latency = 25 milliseconds

Q1. Predict which process will start first?
Q2. Predict which process will finish first?
Q3. Predict which process will finish last?
Q4. Predict how much time process 1 will spend in the Blocked State?
Q5. Predict how much time process 4 will spend in the Run State?

2. Run the simulation to check your predictions. If you were wrong make sure that you understand why?

3. Modify the configuration as follows:

    IO Device Latency = 20 milliseconds

Q6. Predict which process will start first?
Q7. Predict which process will finish first?
Q8. Predict which process will finish last?
Q9. Predict how much time process 2 will spend in the Blocked State?
Q10. Predict how much time process 3 will spend in the Run State?

4. Run the simulation to check your predictions. If you were wrong make sure that you understand why?