School of Information and Physical Sciences
COMP2240/COMP6240 - Operating Systems

Assignment 1 (15%)

Submit using Canvas by **11:59 pm, Sunday 31st August 2025**

1. Introduction

Assignment 1 will focus on your understanding and ability to successfully implement, execute and compare the results of a number of scheduling algorithms.

2. Assignment Task

Write a program that simulates **First Come First Serve (FCFS)**, **Round Robin (RR), Selfish Round Robin (SRR),** and **Feedback (FB)** scheduling algorithms. For each algorithm the program should **list the order and time** of the processes being loaded in the CPU, compute the **turnaround time** and **waiting time** for every process, as well as the **average turnaround time** and **average waiting time** for the set of processes. The average values will be consolidated in a table for easy comparison (*check the sample outputs*).

Two sample input data sets and the corresponding outputs have been supplied. Beyond the two sample datasets provided, additional datasets will be used to test your program. The format of the input data will be the same as in the supplied sample files.

2.1 Input Data

Each input data set contains the following information (*check the sample input files for exact format*):

1. Time for running the dispatcher (`DISP:`) which can be zero or a positive integer
2. For each process:
    i. process id (`PID:`) – as a *string* in the form `p`*n* where n is a positive integer 1, 2, 3, … (in order).
    ii. arrival time (`ArrTime:`) – as a non-negative *integer* representing *unit of time*
    iii. service time (`SrvTime:`) – as a positive *integer* representing *unit of time*

It can be assumed that process P*n* will always arrive before or at the same time of process P*n+1*; times can be considered to just be whole number *'time units'*.

2.2 Dispatcher

It is assumed that the dispatcher is executed to select the next process to run. The dispatcher should behave as follows:

i. The time to run the dispatcher (context switching time) is fixed and taken as input (`DISP:`) from the input file. No other time is wasted in switching between processes other than this.

ii. Whenever a process finishes its time quantum but hasn't completed its service, it goes back to the ready queue and the dispatcher must run again even if there is no other process in the ready queue. For example, in RR scheduling, if process `P1` is running in the CPU and no other process is waiting

in the ready queue, `P1` will be interrupted and sent back to ready queue after its time quantum expires. Then the dispatcher will run again to dispatch `P1` from the ready queue.

iii. If the dispatcher starts at ***t1*** and finishes at ***t2*** (*i.e. time to run the dispatcher is t2-t1*) then in that run, it will choose from the processes that have arrived at or before ***t1***. It will not consider any process that arrives after ***t1*** for dispatching in that run.

iv. If a process `p1` is interrupted at *t1* and another process `p2` arrives at the same time *t1* then the newly arrived process `p2` is added in the ready queue first and the interrupted process `p1` is added after that.

v. If two processes `px` and `py` have all other properties same (e.*g. arrival time, etc.*) then the tie between them is broken using their ID; *i.e.* `px` will be chosen before `py` if $x<y$.

3.1. Scheduling Algorithm Details

Some details about the scheduling algorithms are as follows:

FCFS: Standard FCFS scheduling algorithm.

RR: Standard RR scheduling algorithm with time quantum of 3 ms (or time unit).

SRR: SRR Scheduling is a variant of the canonical Round Robin (RR) scheduling in which each process has its own time quantum q. Each process starts with q = 3 ms and q increases by 1ms each time the process goes through the round robin queue, until it reaches a maximum of 6 ms. Thus, long jobs get increasingly longer time slices.

FB: Standard multi-level FB scheduling algorithm with constant time quantum of 3 ms for each level. A 4 level priority (0 is highest priority and 3 is lowest priority) ready queue is used. The bottom queue (*i.e. the lowest priority queue*) is to be treated in round robin fashion. No priority boosting is used.

3. COMP6240 Additional Task [NOT for COMP2240 students]

In addition to the above simulations, you need to write a report on the CPU scheduling algorithms used in modern operating systems for mobile devices.

Objective: The objective of this assignment is to conduct a detailed research and analysis of the CPU scheduling algorithms used in major mobile operating systems, including Android, iOS, and another contemporary mobile OS of your choice (e.g., HarmonyOS). The emphasis will be on understanding mobile-specific design choices, such as energy-aware scheduling, handling of heterogeneous multi-core architectures (e.g., big.LITTLE), responsiveness for interactive applications, and thermal management. For each OS, you should investigate the specific scheduling algorithm(s) used, their adaptation from desktop counterparts (if applicable), key features, handling of different task types (foreground vs. background, real-time vs. non-real-time), load balancing and scalability in multi-core environments, and mechanisms for achieving fairness, efficiency, and battery optimisation. Your report should demonstrate a deep understanding of how these schedulers are tailored to the constraints and performance goals of mobile devices.

Description: In this assignment, you will investigate and analyse the CPU scheduling algorithms implemented in three major mobile operating systems, specifically Android, iOS, and one other mobile OS of your choice. Your report should provide a comparative analysis of the unique approaches each operating system takes to CPU scheduling in the mobile context, with particular attention to the impact of scheduler design on energy efficiency, system responsiveness, and user experience. You should highlight how mobile CPU schedulers differ from those in desktop/server environments, discuss the trade-offs made between throughput, latency, and battery life, and evaluate how well each OS meets the demands of modern mobile workloads. Your analysis should be supported by technical details, diagrams, and references to academic literature, system documentation, and empirical performance measurements.

Report Structure and Length: You report should contain the following sections (i) Introduction (ii) CPU Scheduling in major OS for mobile devices (iii) Comparative Analysis (iv) Conclusion and (v) References. The length of the report, covering all the sections mentioned before should be 6~10 pages (A4). The report should be written in a formal academic style, with proper citations and references.

The report should be prepared based on your own research and any kind of plagiarism including use of Generative AI in preparing this report should be considered as a serious breach of academic integrity.

4. Submission Requirements

Your submission must also conform to the follow requirements.

4.1. Programming Language

The programming language is Java, versioned as per the University Lab Environment (***Java 21***). You may only use standard Java libraries as part of your submission.

4.2. User Interface

The output should be printed to the console, and strictly following the output samples given in the assignment package. While there are no marks allocated specifically for the Output Format, **there will be a deduction when the result values and result format vary** from those provided.

4.3. Input and Output

Your program will accept data from an input file of name specified as a command line argument. The sample files `datafile1.txt` and `datafile2.txt` (*containing the set1 and set2 data*) are provided to demonstrate the required input file format. **Hint:** the **Java Scanner Library** is something you will likely want to use here!

Your submission will be tested with the above data and will also be tested with other input files.

Your program should output to standard output (*this means output to the Console*). Output should be strictly in the order **FCFS**, **RR**, **SRR**, **FB**, **Summary** (*see the output files*).

The sample files `datafile1_output.txt` and `datafile2_output.txt` (*containing output for* `datafile1.txt` *and* `datafile2.txt` *respectively*) are provided to demonstrate the required output (*and input*) format which **must be strictly maintained**. **If output is not generated in the required format then your program will be considered incorrect.**

Two Gantt charts are provided to explain the corresponding behaviour of different algorithms and the dispatcher for the two sample data files. You are NOT required to generate the Gantt charts.

4.4. Use of Generative AI

For completing the programming component of this assignment, you are encouraged to use Generative AI (GenAI) tools. **But you must NOT use GenAI in preparing the reports**. Please check the document "Use of Generative AI in Assignment 1" for information, explanation and possible use cases on how such tools to be used in preparing the solution before you start working on this assignment.

4.5. Mark Distribution

Mark distribution can be found in the assignment draft marking breakdown documents: `25s2_Assign1_Feedback_2240.pdf` & `25s2_Assign1_Feedback_6240.pdf`.

4.6. Deliverable

The following must be observed in your deliverable:

1. Your submission will contain your program source code with documentation and the report (*below*) in the root of the submission. These files will be zipped and submitted in an archive named **`c9876543.zip`** (*where* **`c9876543`** *is your student number*) – do not submit a `.rar`, or a `.7z`, or etc.
2. Your main class should be **`A1.java`** your program will compile with the command line **`javac A1.java`** and your program will be executed by running **`java A1 input.txt`**.

…where `input.txt` can be *any* filename – **do not hardcode filenames or extensions**!

Note: If your program can not be compiled and executed in the expected manner (*above*) then please add a `readme.txt` (containing any special instructions required to compile and run the source code) in the root of the submission. Please note that such non-standard submissions will be **marked with heavy penalty**.

3. You are required to write a concise **1-page** (A4) report (name as **Report.pdf**) that reviews and reflects on the results obtained from your implementation of the scheduling algorithms in this assignment. The purpose of this report is to analyse and compare the relative performance of the algorithms based on your own implementations, providing explanations for the results you observed. In your discussion, comment on whether your results aligned with theoretical expectations and, if not, suggest possible reasons for any discrepancies. Highlight any interesting trends, patterns, or anomalies, and describe any challenges you encountered during implementation or testing, along with how you addressed them. Your mark for this component will be based on the clarity, quality, and depth of your analysis, as well as the insightfulness of your observations.
4. **[If you have used Generative AI only]** You should write a **2 page** (A4) reflective report (name as **GenAIReport.pdf**) on the use of Generative AI (GenAI) in competing this assignment. In your reflective note, you should provide a detailed account of your experience and should include, but not limited to, the following elements – (i) overview of the tool(s) used (ii) how the tool(s) was/were integrated in your workflow (iii) what was the quality/correctness of the response generated by the tool(s) and/or how the GenAI tool(s) improved the quality of your solution (iv) how the tool(s) improved your learning experience and problem solving skills (v) what challenges you faced in using those tools and how you solved those challenges (vi) how useful was the tool for your productivity e.g. how much time was saved by using GenAI tool(s) (vii) any ethical consideration or concerns in using GenAI tools in this assessment etc. **GenAI must not be used in preparing this report.**
5. **[COMP6240 students only]** You should submit the research report (name as **Research_Report.pdf**) on the task in Section 3. **GenAI must not be used in preparing this report.**

4.7. Submission Notes

1. Assignments submitted after the deadline (**11:59 pm 31st August 2025)** will have the maximum marks available reduced by 10% per 24 hours.
2. If your assignment is not prepared and submitted following above instructions, then you will lose most of your marks despite your algorithms being correct.
3. If you have been granted an extension, please include a copy of the extension approval in as a **PDF document**, called `Extension.pdf`, and place this in the root of your submission, along with your other documents and code.