MANCHESTER METROPOLITAN UNIVERSITY
School of Computing, Mathematics & Digital Technology

OPERATING SYSTEMS PORTFOLIO COVER SHEET

Unit:	6G5Z1105 Networks and Operating Systems
Assignment set by:	Dr Alan Crispin
Verified by:	Dr Soufiene Djahel
Moderated by:	Dr Soufiene Djahel
Assignment title:	**Operating Systems Portfolio**
Type: (GROUP/INDIVIDUAL)	Individual
Hand-in format and mechanism:	Submission is online, via Moodle. More information is available in the attached coursework specification.
Deadline:	As indicated on Moodle.

Learning Outcomes being Assessed

Learning Outcome 1: Demonstrate knowledge and correct application of the principles of concurrency.

Learning Outcome 2: Explain and apply the principles of operating systems in a range of practical and written tasks.

Introduction

The Operating Systems portfolio is a competency portfolio of work, which includes in-class tests and a programming task. The Operating Systems portfolio is marked out of 100 and is worth 50% of your overall mark for this unit.

The breakdown of work in this portfolio is:

Portfolio Element	Marks	Schedule
In-class test MCQ1	30	Covering topics: Overview, processes, Linux operating system tools, CPU scheduling
Programming task	40	Analysis of C scheduling algorithms
In-class test MCQ2	30	Covering topics: System calls, file systems, memory management, virtual memory, virtualization

In Class Tests

In-class tests will be in the form of Moodle quizzes. The quizzes will be based on materials covered in both the lectures and lab sessions. It is important that you study the laboratory materials (including screencasts) in preparation for the tests. Further details and support on how to prepare for the tests will be given in lecture sessions.

Programming Task

The aim of the programming task is for you to demonstrate your knowledge of CPU scheduling algorithms. The detailed specifications of this programming task are stated below. You should read the specification carefully, and make sure that your code conforms to the specifications. You should also develop the habit of documenting your code as you write it. You will not be able to achieve full marks for this section of the task if your code is not adequately documented.

There are two weeks of lab classes dedicated to C programming relating to this task, but you should also dedicate time outside class to working on this task. During your scheduled classes, you should show your work to the staff member in the lab, who can provide formative assessment of your work.

Programming Task -Details

Process scheduling is an important component of Operating Systems process management. There are many algorithms for process management such as:

1. First Come First Served (FCFS)
2. Shortest Job First (SJF)
3. Round Robin (RR) scheduling.

The programming task is to design and develop a C console program for simulating a First Come First Served (FCFS) algorithm, a Shortest Job First (SJF) and a Round Robin Scheduling algorithm on a single CPU and analyse the performance of each algorithm by calculating performance metrics (turn around time and wait time). Your program is required to print out a scheduling table such as that shown in Fig.1 for the FCFS algorithm (you can add column and row lines to the table)

```
FCFS CPU Scheduling Algorithm
AT      BT      TaT     WT
  0       4       4       0
  1       5       8       3
  2       2       9       7
  3       1       9       8
  4       6      14       8
  6       3      15      12
Average Turn Around Time: 9.833333
Average Wait Time:6.333333
```

Fig.1. Example scheduling table for the FCFS algorithm where AT =arrival time, BT=burst time, TaT = turn around time and WT=wait time.

You algorithms should be tested with the following process data where “PID” is the process ID number, AT is the arrival time to the CPU and BT is the burst time.

Assignment Dataset1

PID	AT	BT
0	0	3
1	1	6
2	2	8
3	3	25
4	4	5
5	6	20

Assignment Dataset2

PID	AT	BT
0	0	25
1	1	20
2	2	3
3	3	8
4	4	6
5	6	5

Assignment Dataset3

PID	AT	BT
0	3	4
1	1	5
2	2	20
3	0	25
4	6	14
5	8	6

Fig. 2. Assignment datasets to be used for testing scheduling algorithms

First Come First Served (FCFS) Algorithm

The FCFS algorithm has the following features:

1. It is a non pre-emptive algorithm (i.e. a process runs until finished)
2. A process is selected based on its arrival time to the ready queue

Shortest Job First (SJF) Algorithm

The SJF algorithm has the following features:

1. It is a non pre-emptive algorithm (i.e. a process runs until finished)
2. A process (among the available processes) is selected based on shortest remaining time to complete

Round Robin (RR) Algorithm

The Round Robin (RR) algorithm has the following features:

1. A First In First Out (FIFO) ready queue is maintained for processes to be scheduled onto the CPU
2. A fixed time is allocated to every process that arrives in the queue which is known as the time slice or time quantum
3. The first process that arrives in the ready queue is selected and sent to the CPU for execution. If it is not able to complete its execution within the time quantum provided it is stopped and sent to the back of the ready queue. The state of the process is saved (context switching) so that it can resume from the point from where it was stopped
4. The scheduler then selects the next process in the ready queue and dispatches it to the CPU for execution for the given time quantum
5. These steps are repeated until all the processes have finished.

Your C program should implement a console menu system such as that shown below to allow each of the scheduling algorithms to be selected and tested.

```
1 - First Come First Served (FCFS) Scheduler
2 - Shortest Job First (SJF) Scheduler
3 - Round Robin (RR) Scheduler
4 - Quit
Enter a selection: 2
```

Fig. 3. Example menu system

You will be rewarded for incorporating file read/write code so that performance analysis can be done using an input file containing process data and scheduling metrics can be saved to an output file.

Deliverables (Programming Task Submission)

You are required to submit the following deliverables.

1) A complete copy of your source code (fully commented-see below)
2) Screenshots of the scheduling results for the FCFS, SJF and RR algorithms undertaken using the assignment datasets
3) A completed assignment template form showing the features that you have completed

You should submit a complete zipped copy of the above to Moodle with the format

surname_studentnumber.zip

Please ensure that all code submitted is compiled using gcc (The GNU C compiler) and runs at the Linux command line and your name and student number are included as comments in your source code files.

Commenting Code

It is important that you fully comment your source code to document the meaning and purpose of your source code, specifically explaining how your scheduling algorithms work. Comments provide clarity to the C source code and help in debugging the code. List the advantages and disadvantages of each scheduling algorithm in your comment section by reflecting on the results with each of the datasets.

You can create a comment that spans multiple lines. For example:

```
/*
 * Author:          Alan Crispin
* Purpose:         The purpose of this function is to ...
* Explanation:     This function .. explain how it works..
*  Parameters:     List and explain the input parameters
*  Returns:        List and explain the return type
*  Advantages:
*  Disadvantages:
 */
```

The compiler will assume that everything after the /* symbol is a comment until it reaches the */ symbol, even if it spans multiple lines within the C program. A double slash comment prefix // can be used to comment single lines.

Code Quality

Your work will be assessed on code quality which can be defined in terms of: Does the code do what it is supposed to do? Does it contain defects or problems? Is it easy to read, maintain and extend?

Some tips for best coding practice include:

1) Name functions and variables wisely and succinctly so that they make sense when others read the code
2) Avoid using global variables as these can be accessed and manipulated anywhere in the code making maintenance a nightmare (use constants)
3) Always attempt to write reusable code
4) Ensure the code is easy to read and understandable

Choose and stick to a style for naming various elements of the code. This helps with understanding the logical flow. Ensure code is formatted (indented where appropriate). Consistency is more important than a specific formatting style.

Test your code under worst case conditions. Always compile with warnings on -Wall (Warning all) as stable code is free from errors and has no warnings.

Assignment Template Form

The assignment template form is shown below and can be downloaded from Moodle and requires you to complete the following.

Assignment Functionality Completed:

FCFS scheduler

☐ First Come First Served (FCFS) scheduler code partially implemented and tested
☐ First Come First Served (FCFS) scheduler code fully implemented (sorting) and tested

SJF scheduler

☐ Shortest Job First (SJF) scheduler code partially implemented and tested
☐ Shortest Job First (SJF) scheduler code fully implemented and tested

Round Robin scheduler

☐ Round Robin (RR) scheduler code partially implemented and tested

☐ Round Robin (RR) scheduler code fully implemented and tested

Console menu

☐ Application has a menu system to allow different scheduling algorithms to be selected and tested from the console partially implemented

☐ Application has a menu system to allow different scheduling algorithms to be selected and tested from the console fully implemented

File read/write

☐ File read/write code to load datasets and store results

Code commented

☐ Code is well presented with occasional comments

☐ Code is well presented with sensible high quality comments explaining algorithms

☐ Comments incorporated on advantages and disadvantages of schedulers

Screenshots

☐ Screenshots of FCFS scheduler (3 assignment datasets))

☐ Screenshots of SJF scheduler (3 assignment datasets)

☐ Screenshots of RR scheduler (3 assignment datasets)

Additional Comments

Please indicate any features that have been partially implemented or any other issues that need to be drawn to the attention of the markers.

Screencasts: to support this assignment include those on First Come First Served (FCFS), Shortest Job First (SJF), Round Robin (RR) and Shortest Remaining Time First (SRTF) scheduling.

Feedback: Marks will be made available via Moodle with a turnaround time in accordance with University guidelines. Written feedback will be provided on Moodle to each student by tutors.

Mark Scheme: The programming task will be graded in a number of areas, attracting a number of marks for each. Guidance on the assessment criteria for each area is included below. Note that in each area you must work upwards in terms of features implemented to achieve the maximum mark for that area.

First Come First Served (FCFS) scheduler		**Shortest Job First (SJF) Scheduler & menu system**		**Round Robin Scheduler**		**Code Comments/Screenchots**	
FCFS scheduler implemented and tested	15 marks	Evidence of an attempt at menu system and SJF scheduler partially implemented and tested	10 marks	Round Robin scheduler partially implemented and tested and incorporated into menu system	10 marks	Code is **poorly** or **moderately** presented, with **occasional** comments and assignment template completed and submitted	2 marks
						Code is **well** presented, with **sensible** comments covering advantages/disadvantages of scheduling algorithms	9 marks
FCFS scheduler fully implemented allowing processes to be sorted according to arrival time and tested with all assignment datasets with performance metrics calculated and printed	10 marks	SJF scheduler fully implemented and incorporated into a console menu system with performance metrics calculated and printed	15 marks	Round Robin scheduler fully implemented with performance metrics calculated and printed and incorporated into menu system	15 marks	Screenshots for each algorithm for each dataset (3x3)	9 marks
				File read/write code	5 marks		
Section Total	25		25		30		20

For example, a student earning 25 marks in the *FCFS section*, 15 marks in *SJF section*, 10 marks in *Round Robin section*, 10 marks for c*ode quality* and *5* marks for the *screenshots of scheduling results* would achieve an overall mark of 65% (25 + 15+ 10+ 10+5) for this element of the portfolio. If you need any help in understanding this assignment please talk to the tutor who takes you for your laboratory sessions or arrange to see Dr Alan Crispin.