Faculty of Information Technology

Electrical and Computer Engineering Department

OPERATING SYSTEMS (ENCS3390)

Project Report

Instructor:	Dr. Ayman Hroub	
Partners:	Diyar Barham	1191314
	Al-Ayham Maree	1191408
	Sara Ammar	1191052

Section:	3 & 4
Report Deadline:	9-Jan-2022

Abstract

Second Chance and Clock, the even ones in the algorithms given in the project description (2,4) depending on the max value of the least significant digit of the team ID numbers, were implemented in this project as a simulator for experimenting with page replacement methods with threading and Scheduling. We also created a user interface that is both straightforward and comfortable to use.

Table of Contents

Table of Figures

Theory

A FIFO replacement algorithm is the foundation of the second-chance replacement algorithm. However, after a page has been identified, we examine its reference bit. If the value is 0, we will replace this page; however, if the value is 1, we will give the page a second opportunity before moving on to the next FIFO page. A page's reference bit is cleared and its arrival time is reset to the current time when it receives a second opportunity. As a result, a second-chance page will not be changed until all other pages have been replaced (or given second chances). Furthermore, a page will never be changed if it is utilized frequently enough to maintain its reference bit set.

A circular queue is one approach to implement the second-chance algorithm (also known as the clock algorithm). The next page to be replaced is indicated by a pointer (a hand on the clock). The pointer advances until it reaches a page with a 0-reference bit when a frame is required. It clears the reference bits as it progresses. When a victim page is located, it is replaced, and a fresh page is inserted in its place in the circular queue. When all bits are set in the worst scenario, the pointer cycles over the whole queue, giving each page a second opportunity. Before picking the next page for replacement, it clears all the reference bits. If all bits are set, second-chance replacement degenerates into FIFO replacement.

Within a process, a thread is a path of execution. Multiple threads can exist in a process. The lightweight process is also known as a thread. By splitting a process into many threads, concurrency may be achieved.

Process Scheduling is a job in the operating system that schedules processes in various stages such as ready, waiting, and running. Process scheduling allows the operating system to assign each process a time period for CPU execution. Another significant advantage of employing a process scheduling system is that it keeps the CPU active at all times. This enables you to get the quickest possible reaction time for applications.

Procedure

Implementation:

We utilized the Java programming language in our implementation. Main Class, Memory Class, TableData Class, Process Class, Memory Table class, and Result Table class are the java classes we created. Memory Class was like a single memory cell, consisting of a page and its bit reference, and so the memory was an array of Frames, Process Class was like a single memory cell, consisting of the process attributes (burst time, arrival time, start time, finish time, etc...), and so processes were java objects from this class, TableData class was like a single memory cell, consisting of the page number and the pid of the process that owned this page, and finally, Main class was the class that holds all the processing of the data and functions. Result Table class are the java classes were for the GUI.

Running Our Program

Our interface is very easy to use, as shown in the following screenshot, you just need to select a file from the computer then click the "Read" button to input the data. After that you need to input memory size and time quantum, select an algorithm and click "Run" to run the program.

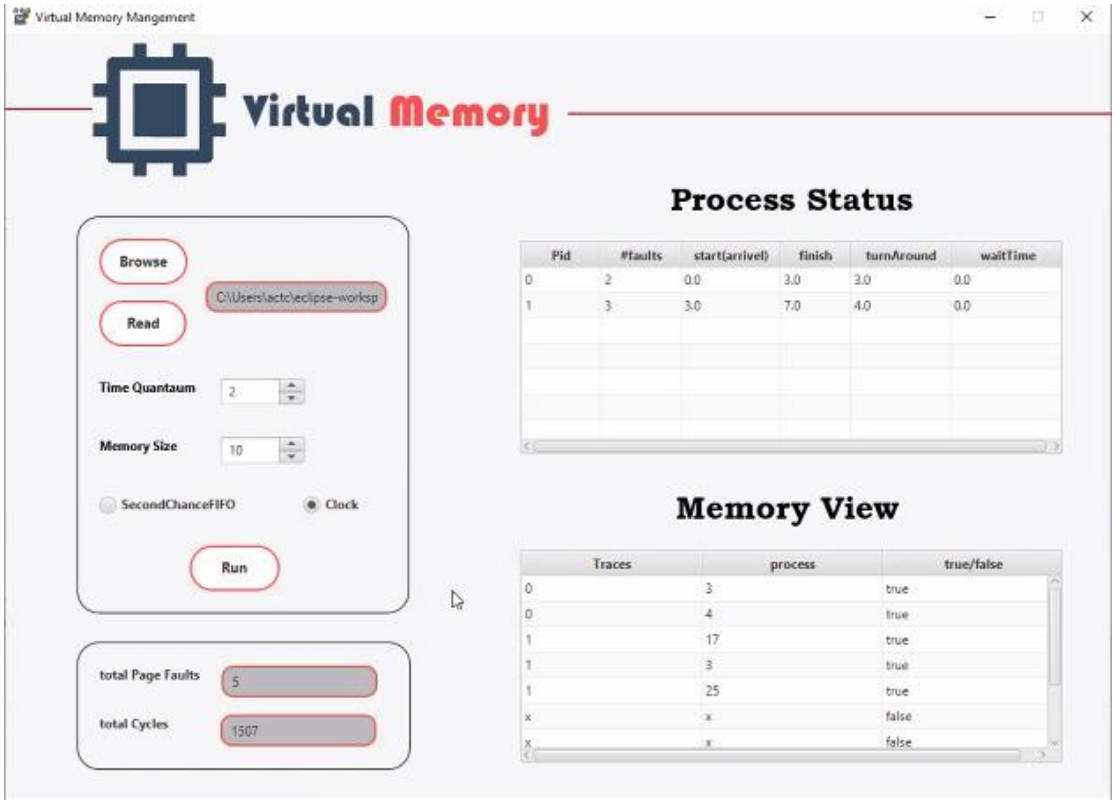

Figure 1: Program Interface

Input Files Structure

The structure of our program's input file should be similar to the following screenshot. The first three are used as the same way the input file structure format decided in the project file, the fourth and fifth lines will be skipped (but are required for our code), and are just included to make it easier to enter and read data from the file. The "|" separator is required between each line's numbers (but the spaces are not, you can enter data without spaces). And at the end of each line the traces in hexadecimal are separated by commas ",", and in the program each trace first 12 lower bits will be deleted to produce a page number.

```
2
10
2
PID      Start      Duration        Size in pages      Memory traces
.....................................................................................................
0    |      0    |       3       |       3          |00003fff,00004fff,00003fff
1    |      5    |       4       |       4          |00011efa,00003fad,00011daa,00019aab
```

Figure 2: Input Files Structure

Test Case 1

```
2
10
2
PID      Start      Duration        Size in pages      Memory traces
.....................................................................................................
0    |      0    |       3       |       3          |00003fff,00004fff,00003fff
1    |      5    |       4       |       4          |00011efa,00003fad,00011daa,00019aab
```

Figure 3: Small Input Data File

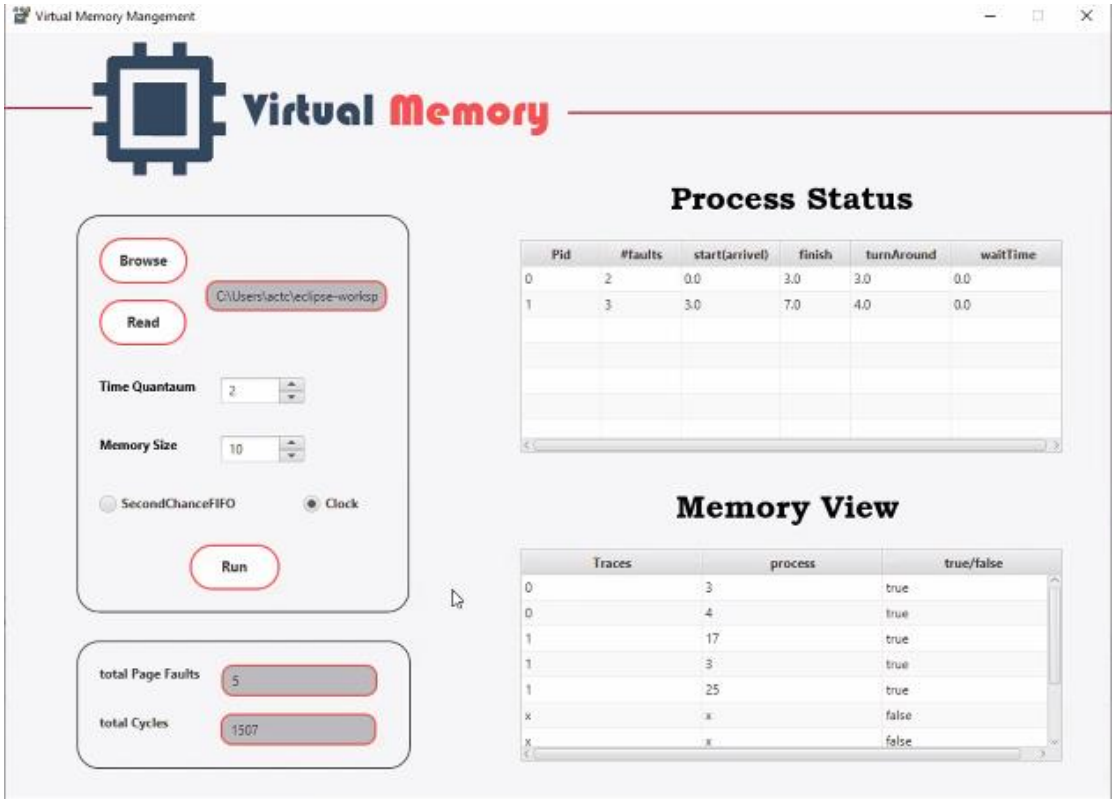

Figure 4: Small Input Clock Result

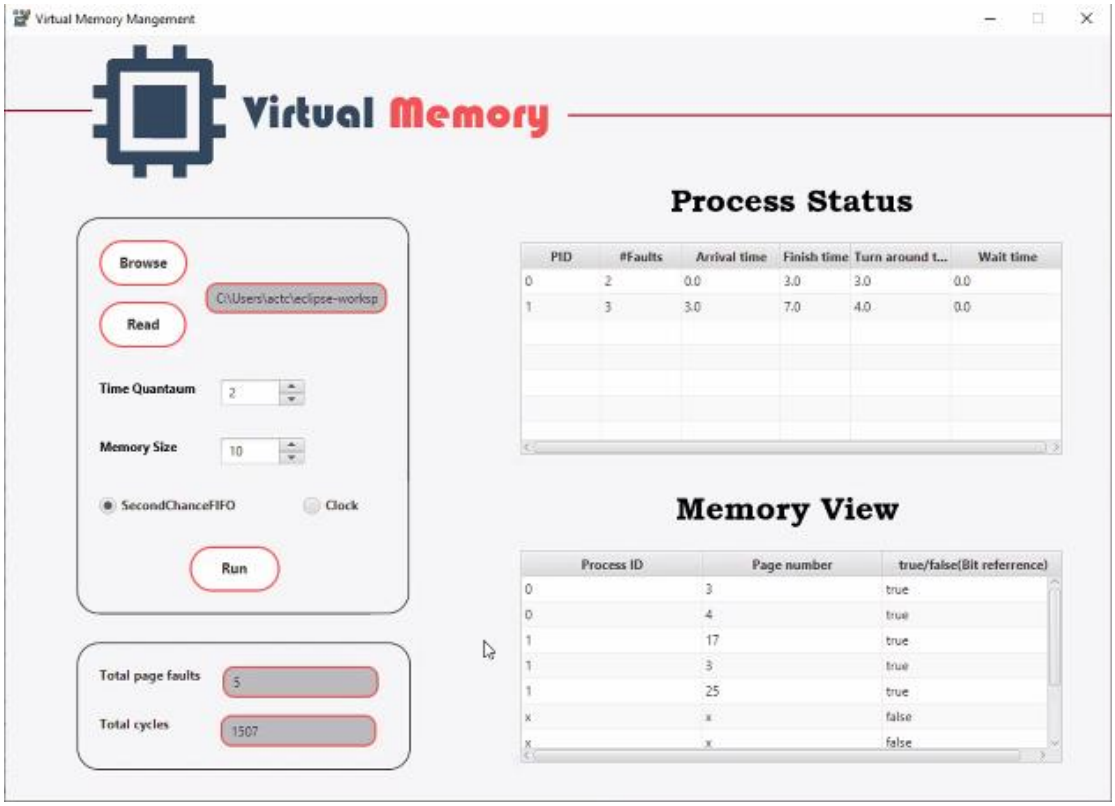

Figure 5: Small Input Second Chance FIFO Result

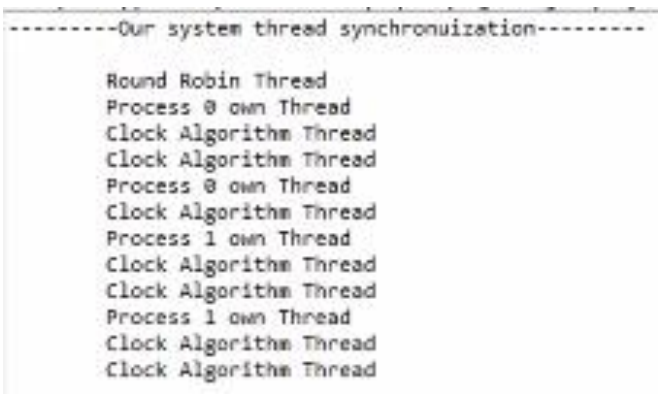

Figure 6: Small Input Threading Results

Test Case 2

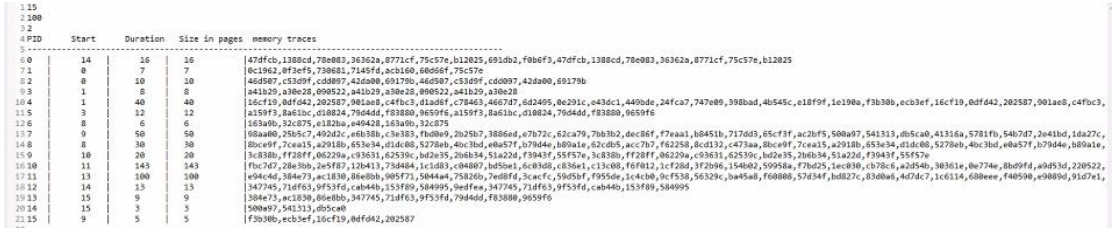

Figure 7: Big Input Data File

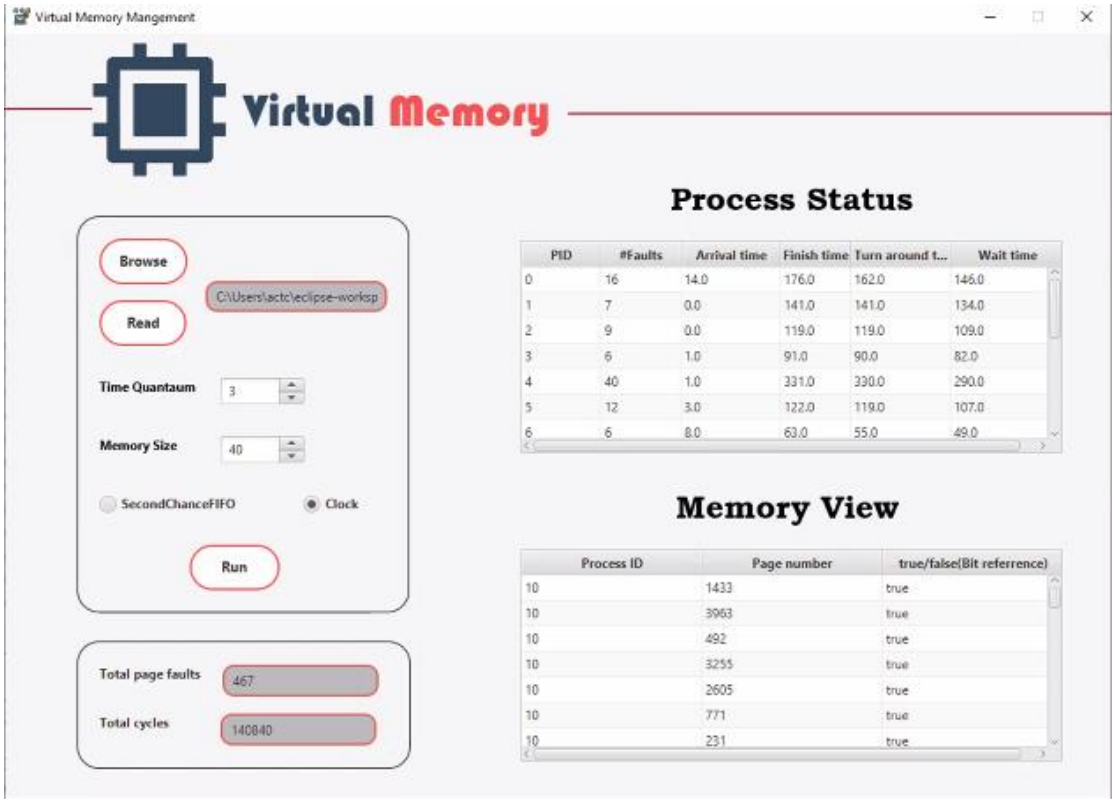

Figure 8: Big Input Result in Clock and 40 Memory Size

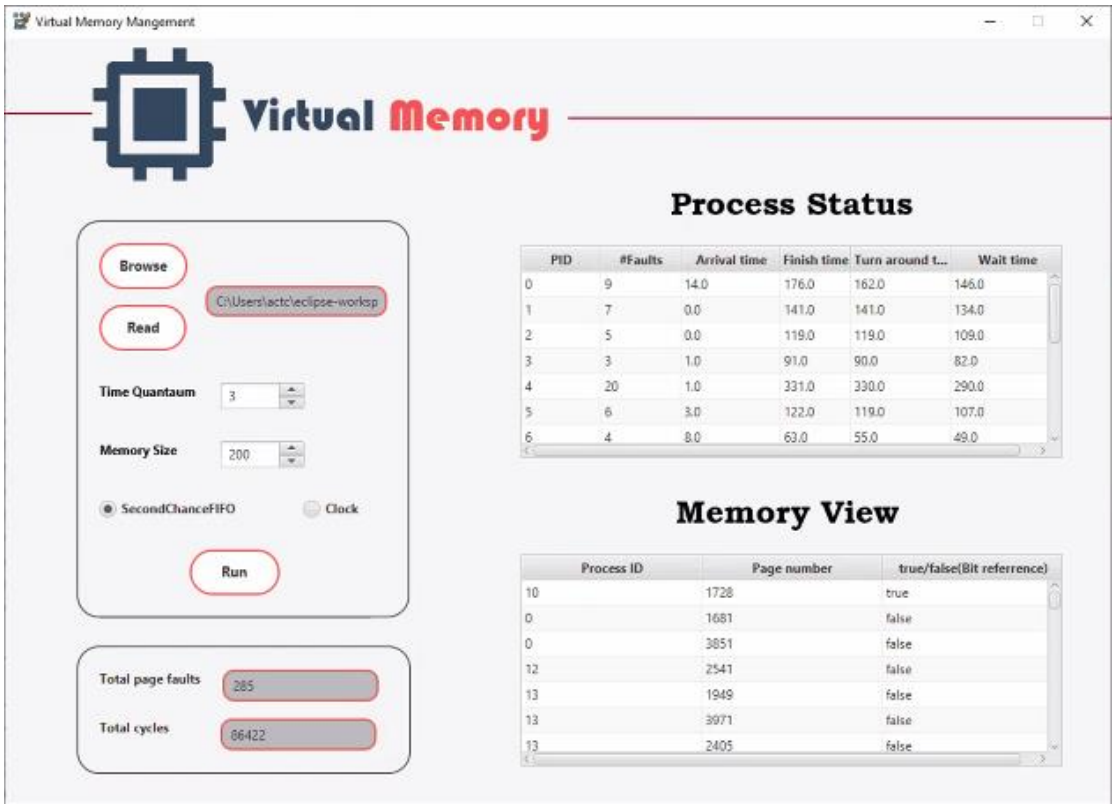

Figure 9: Big Input in Second Chance FIFO and 200 Memory Size

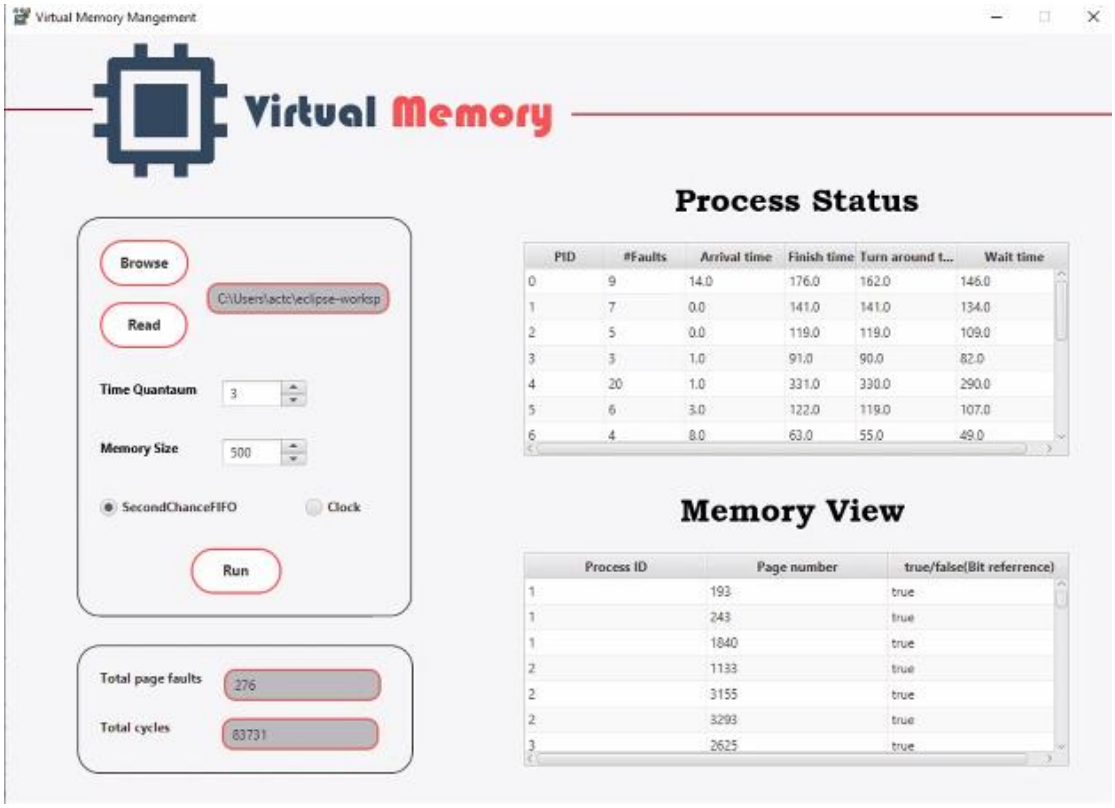

Figure 10: Big Input in Second Chance FIFO and 500 Memory Size

Conclusion

This project allows users to view and test several page replacement methods, compare them, and determine which is best for a certain purpose, as some algorithms may produce fewer page faults than others, while others may be better in other situations. Simulating algorithms is beneficial because it allows us to evaluate their efficiency before putting them into practice, and it was great to have a project to teach us how to do so. The threading part was useful to understand how threading work and how to implement it but a downside of the project is that threading doesn't change the output so we couldn't know if it was working or not without debugging.

References

- Operating System Concepts, 10th Edition.
- Geeks for Geeks, [Link].
- Baeldung, [Link].
- Stack Exchange, [Link].
- EXAMRADAR, [Link].
- JENKOV, [Link].
- Guru99, [Link].
- Geeks for Geeks, [Link].