CS 4310 Operating Systems
Project #2 Simulating Page Replacement Manager and Performance Analysis

Due: 12/3
(Total: 100 points)

Student Name: ____

Date: __________________________________

Important:
- Please read this document completely before you start coding.
- Also, please read the submission instructions (provided at the end of this document) carefully before submitting the project.

Project #2 Description:

Simulating Page Replacement Manager of the Operating Systems by programming the following three page replacement algorithms that we covered in the class:

a. First In First Out (FIFO)
b. Least Recently Used (LRU)
c. Optimal Algorithm (OPT)

You can use either Java, or C++ for the implementation. The objective of this project is to help student understand how above three page replacement algorithms operates by implementing the algorithms, and conducting a performance analysis of them based on the performance measure of ***page faults*** for each page replacement algorithm using multiple inputs. Output the details of each algorithm's execution. You need to show what pages are inside the page frames along with the reference string and mark it when a page fault occurred. You can choose your display format, for examples, you can display the results for each reference string in a table format as shown in the class notes. The project will be divided into three phases to help you to accomplish above tasks in in a systematic and scientific fashion: Design and Testing, Implementation, and Performance Analysis.

The program will read in a testing data file consisting 50 reference strings of length 30 (TestingData.txt) – this file will be generated by you. In this project, assume that

(1) the length of the reference string is always 30, e.g. 361724720354720146353214567012.
(2) there are 8 pages, from 0 to 7.

You are required to run this testing data file using the three page replacement algorithms with four different page frame sizes (3, 4, 5 and 6).

You can structure the testing data in your own way.
A look of a sample testing data file, named "TestingData.txt", is given as follows:

```
[Begin of TestingData.txt]
361724720354720146353214567012
726301625203767261726562516250
…
342514361524372635435261547265
[End of TestingData.txt]
```

You can implement the algorithms in your choice of data structures based on the program language of your choice. Note that you always try your best to give the most efficient program for each problem.

Submission Instructions:

- ***turn in the following @canvas.cpp.edu after the completion of all three parts, part 1, part 2 and part 3***
    - ***(1) testing data file (with 50 reference strings of length 30)***
    - ***(2) three program files (your choice of programming language with proper documentation)***
    - ***(3) this document (complete all the answers)***

Part 1
Design & Testing (30 points)

a. Design the program by providing pseudocode or flowchart for each page replacement algorithm.

<insert answers here>

b. Design the program correctness testing cases. Give at least 4 testing cases (with 3, 4, 5, or 6, page frames) to test your program, and give the expected correct output (# of page faults) of the program for each case in order to test the correctness of each algorithm.

<complete the following table>

Testing case #	Input reference string	Expected # of page faults for FIFO (√ if Correct after testing in Part 3)	Expected # of page faults for LRU (√ if Correct after testing in Part 3)	Expected # of page faults for Optimal Algorithm (√ if Correct after testing in Part 3)
1 (3 page frames)	<insert reference string answers here>	<insert answers here>	<insert answers here>	<insert answers here>
2 (4 page frames)	<insert reference string answers here>	<insert answers here>	<insert answers here>	<insert answers here>
3 (5 page frames)	<insert reference string answers here>	<insert answers here>	<insert answers here>	<insert answers here>
4 (6 page frames)	<insert reference string answers here>	<insert answers here>	<insert answers here>	<insert answers here>

c. Design testing strategy for the programs. Discuss about how to generate and structure the randomly generated inputs for experimental study later in Part 3.

To study the performance of the three page replacement algorithms, let's use a random number generator for generating the testing data set file of 50 reference strings of length 30 as the input for the programs. However, student should use this same data set for running each of the three page replacement algorithms and each of the four frame sizes (3, 4, 5, 6).

The average performance (average number of page faults) can be calculated after an experiment is conducted using the testing data set file of 50 reference strings of length 30.

For example, if $X = \{x_1, x_2, x_3 \ldots x_{50}\}$ *contains 50 results (number of page faults) using the testing data set as input for FIFO and page frame size of 3, then*

average number of page faults for FIFO and page frame size of 3 $= \frac{\sum_{i=1}^{50} Xi}{50}$

<answer how you generate and structure the randomly generated inputs for this project>

Part 2
Implementation (30 points)

a. Code each program based on the design (pseudocode or flow chart) in Part 1(a).

   <generate three programs and stored them in three files, needed to be submitted>

b. Document the program appropriately.

   <generate documentation inside the three program files>

c. Test you program using the designed testing input data given in the table in Part 1(b), Make sure each program generates the correct answer by marking a "√" if it is correct for each testing case for each program column in the table. Repeat the process of debugging if necessary.

   <complete the three columns of the three algorithms in the table @Part 1(b)>

d. For each page replacement program, capture a screen shot of the execution (Compile&Run) of one testing reference string to show how this program works properly

   <insert totally three screen shots, your choice of page frame size one for each program, here>

By now, three working programs are created and ready for experimental study in the next part, Part 3.

Part 3
Performance Analysis (40 points)

a. Run each program with the designed randomly generated input data given in Part 1(c). Generate a table for all the experimental results for performance analysis as follows.

number of page frames	Average number of page faults (FIFO Program)	Average number of page faults (LRU Program)	Average number of page faults (Optimal Algorithm)
3 page frames	*<insert average number of page faults result here>*	*<insert average number of page faults result here>*	*<insert average number of page faults result here>*
4 page frames	*<insert average number of page faults result here>*	*<insert average number of page faults result here>*	*<insert average number of page faults result here>*
5 page frames	*<insert average number of page faults result here>*	*<insert average number of page faults result here>*	*<insert average number of page faults result here>*
6 page frames	*<insert average number of page faults result here>*	*<insert average number of page faults result here>*	*<insert average number of page faults result here>*

b. Plot a graph of each algorithm, average page fault vs. page frame size (3, 4, 5, 6) and summarize the performance of each algorithm based on its own graph.

<insert totally three graphs, one for each page replacement program, here>

<write a summary for each graph >

Plot all three graphs on the same graph and compare the performance (page faults) of all three algorithms. Rank three page replacement algorithms. Try giving the reasons for the findings.

<insert a three-graphs-in-one graph here>

< write a summary of your results in this graph>

c. Conclude your report with the strength and constraints of your work. At least 100 words. (Note: It is reflection of this project. If you have a change to re-do this project again, what you like to keep and what you like to do differently in order get a better quality of results.)

<write a conclusion about strength and constraints of your work here.>