@author: Brian Gebel, Abhishek Choudhary

@username: masc0974, masc0980

@classInfo: CS570, Summer 2013

@Assignment # 2, Page replacement algorithms

@filename: README

**Files in the project:**

* makefile
* p2.h
* p2algorithms.c
* p2helper.c
* p2main.c
* README

**Compile Instructions:**

1. make input
2. make

The above two commands will create the pages.txt file and then compile the files in the directory respectively.

**Operating instructions:**

./a2 in order to run the program

This will prompt the user to enter the number of page frames the system has in its main memory.

**List of significant design ideas:**

We separated different routines: process in p1main.c, thread routine in p1thread.c and all helpers in p1heleper.c

Basically, abstracted the different sections of process

No error handling in helper class, it is done in the main and thread routine

**List of extra features:**

* + - 1. Alternatively you can use the cmd “make input2” which will generate another set of numbers.
      2. For file clean up use cmd “make clean” which will remove all object files and remove pages.txt

**List of known bugs:**

N/A

**Lessons learned:**

* Learnt how memory paging works on a system.
* Analyzed the efficiency of the THREE paging replacement algorithms, by comparing their performances.
* We also noticed that the more frames, the less amount of page faults since more frames means more data can be stored in the page tables.

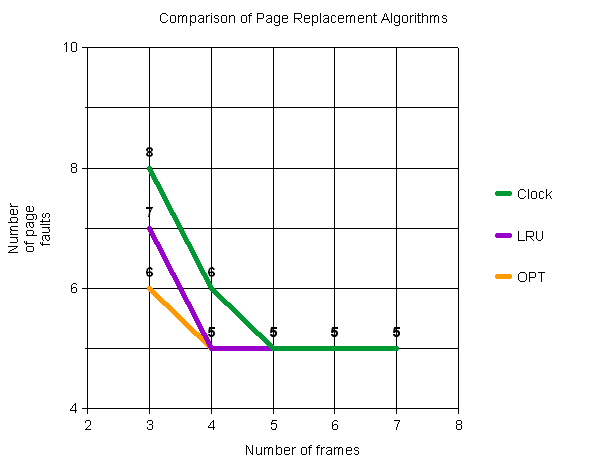
**Summary of the findings:**

After analyzing the number of faults for the different algorithms based on the input stream, we come to the following conclusions:

* Optimal algorithm is always better or as good as the other algorithms on all the data sets. However, as we learnt in the class, this algorithm is unrealistic in real world scenario as it is hard to determine the data coming in and the size of the frame.
* LRU and Clock are almost even in terms of performance depending on the data that is coming in and the frame size

**For page address stream: 2 3 2 1 5 2 4 5 3 2 5 2**

|  |  |  |  |
| --- | --- | --- | --- |
| **No. of page frames** | **Clock**  **Number of page faults** | **LRU**  **Number of page faults** | **Optimal**  **Number of page faults** |
| **3** | **8** | **7** | **6** |
| **4** | **6** | **5** | **5** |
| **5** | **5** | **5** | **5** |
| **6** | **5** | **5** | **5** |
| **7** | **5** | **5** | **5** |



**For page address stream: 5 2 4 5 2 6 8 7 2 9 3 6 9**

|  |  |  |  |
| --- | --- | --- | --- |
| **No. of page frames** | **Clock**  **Number of page faults** | **LRU**  **Number of page faults** | **Optimal**  **Number of page faults** |
| **3** | **10** | **10** | **8** |
| **4** | **10** | **9** | **8** |
| **5** | **9** | **9** | **8** |
| **6** | **8** | **8** | **8** |
| **7** | **8** | **8** | **8** |

