Paging Algorithm Simulation

Prepared by,
Soh Yee Lee
2009 September

Explanation of source code

There is two parts in our display when we simulate the paging algorithm, which is the top part of M and the bottom part of M given in the question. The top part of M, which represents frames in memory, contains all the pages that are in the memory, while the bottom part of M contains some of the pages that have been swapped out. The size of M is not fixed, it is depends on the number of virtual page.

While the process is running, if the require page is found in the frames (the top part of M), the page will be moved to the top of the frames from its current position, while the other page in the frame will move one unit down. On the other hand, if the page is not found in the frames and there is no free frame for a new page in the memory, we need to use page replacement algorithm to find a “victim” page and swap it into a backing store to allocate a new space for the new page in the memory. The victim page will be then removed from the frames (the top part of M) and it will be shown at the first position in the bottom part of M. The new page will be loaded to the memory and it will be shown in the top part in the memory.

The model described on top will be used to show three types of page replacement algorithm as required. The three types of page algorithm are Least-Recently-Used (LRU) algorithm, Least-Frequently-Used (LFU) algorithm and First-In-First-Out (FIFO) algorithm.

For the LRU algorithm, there is 2 ways to perform this algorithm, which is using counters and stack. Here, we will perform this algorithm using stack. LRU algorithm replaces a page that has not been used for longest time. When a page is being referenced, if it is in the memory, it will be removed from the stack and will be putted on the top of the stack. However, if the page is not in the memory and the memory is full, the least used page in the frame will swapped out and the new page will be putted on the top of the stack. The model on the top is quite similar with LRU algorithm, which is the most recently used page, is always on the top of the stack and the least recently used page is always at the bottom. Deque is one of the data structure in C++ that can be used like a stack. An item can be inserted or deleted from any position of deque, including insert an item to the top of the deque. Due to this significance, it is used in the program rather than array because because we need to shift the item in the array if we delete an item in the array.

As for LFU, we maintain a counter for each page in the memory. Whenever a page is referenced, we add one to the page's counter. If a new page comes in and the memory is full, we replace a new page with the page that has least counter in the memory. If there are at least 2 pages with same counter, then we will use FIFO algorithm, which means the oldest page in the queue, will be swapped out. In the program, two deques are used. One is used as queue, and another one is used to keep track the counter for each page in the queue. Deque is used again because of the convenience to insert or delete an item from any position from it.

FIFO algorithm keep tracks which page come in first, and if a page in a memory needs to be replaced, the oldest page in the memory is chosen. An FIFO algorithm is implemented by a queue. When a new page comes, it is inserted to the tail of the queue, and if the queue is full, the head of the queue is popped. In the program, we use the deque as a queue.

For LFU and FIFO algorithm, because the output is not according to its original output (Since the model is using LRU as its standard output), so whenever we replace an old page, we need to perform a search in the frame, find the page that will be swapped out, and remove it from the deque. Then the new page is push to the top of the frame.

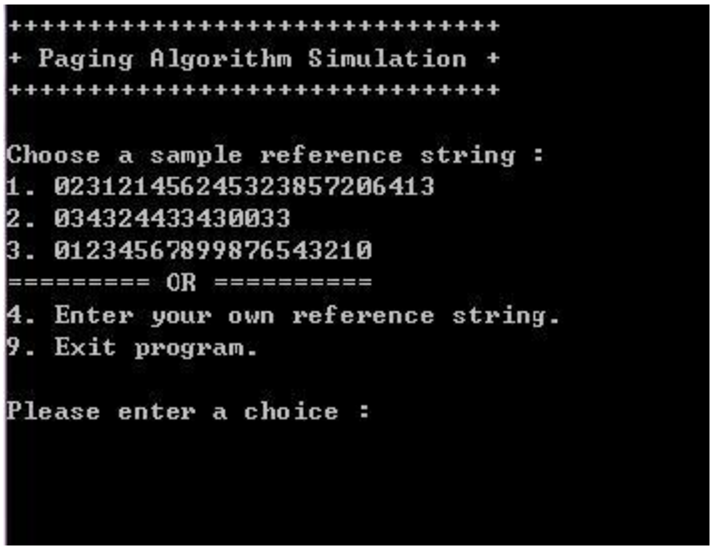
For simulating different number of frame in the memory, the algorithm is same, the difference is just the number of frame, which is the top part of the M and the number of the page that had been swapped out, which is bottom part of the M is changing.

How to run the program:

To compile, type "g++ paging.cpp WinConsole.cpp" in the command prompt.

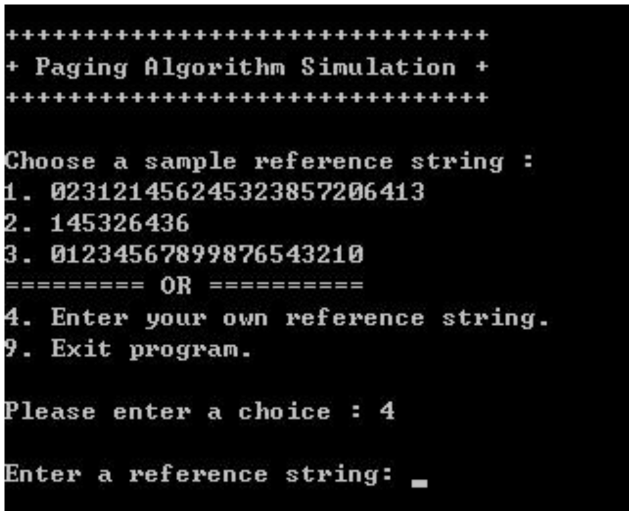
Below are some descriptions on how to use the program.

When the program starts, a main menu is shown:



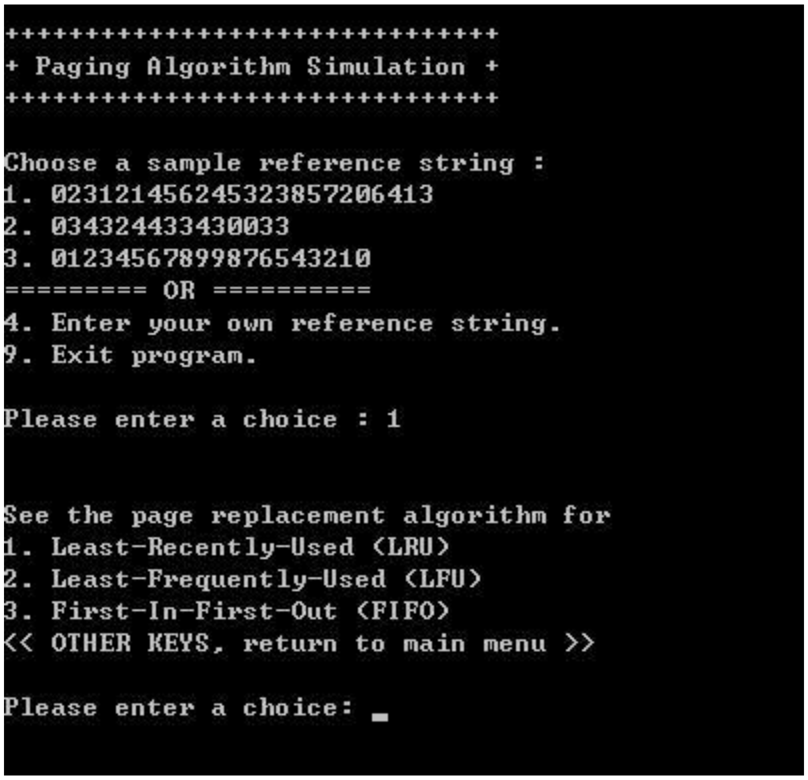
```
++++++++++++++++++++++++++++++
+ Paging Algorithm Simulation +
++++++++++++++++++++++++++++++

Choose a sample reference string :
1. 023121456245323857206413
2. 034324433430033
3. 01234567899876543210
========= OR =========
4. Enter your own reference string.
9. Exit program.

Please enter a choice :
```

Choice 1, 2, 3 are the sample input for the program, if you want to enter your own string, enter 4.



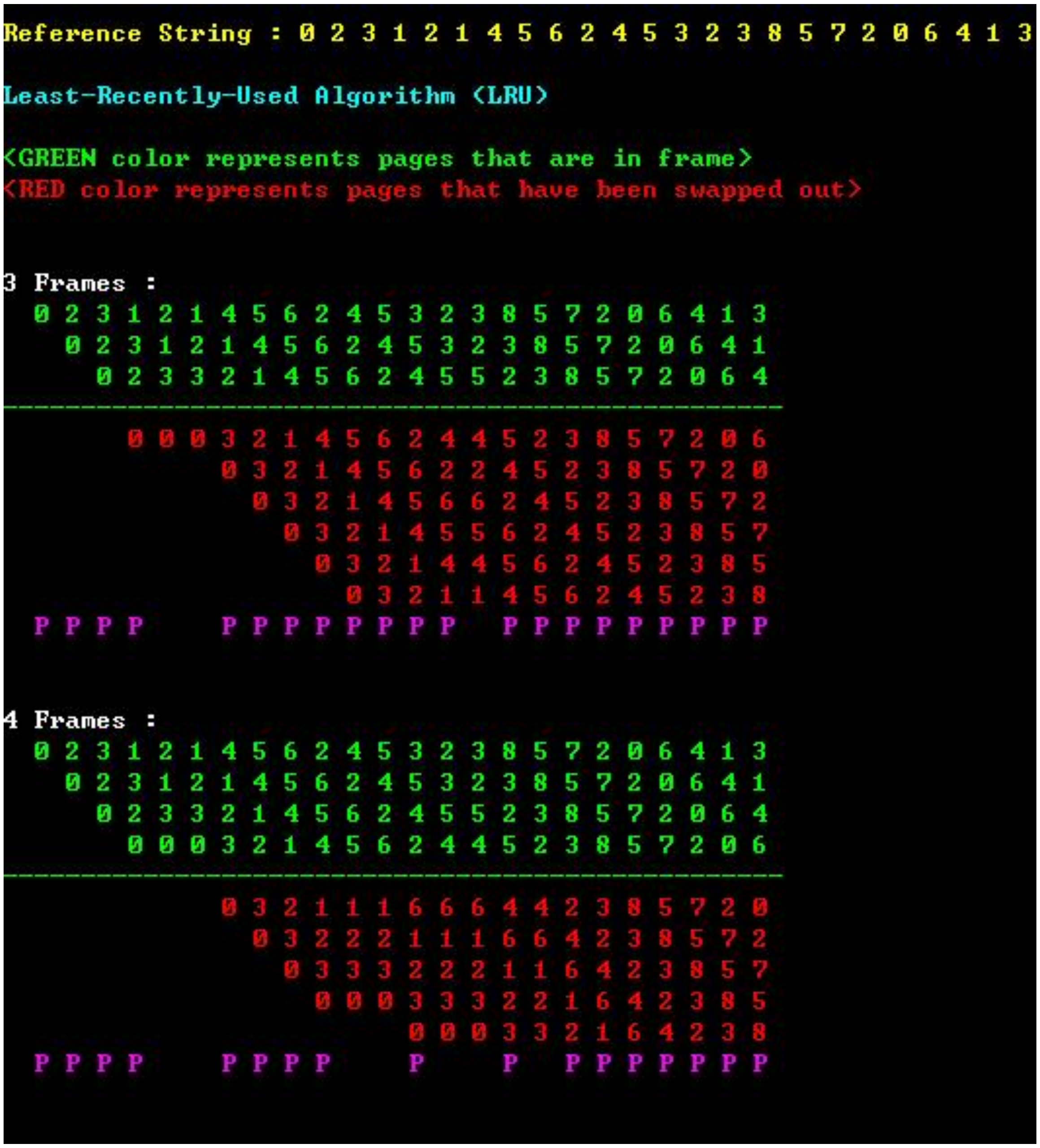
```
++++++++++++++++++++++++++++++
+ Paging Algorithm Simulation +
++++++++++++++++++++++++++++++

Choose a sample reference string :
1. 023121456245323857206413
2. 145326436
3. 01234567899876543210
========= OR =========
4. Enter your own reference string.
9. Exit program.

Please enter a choice : 4

Enter a reference string: _
```

After choosing the reference string, another menu will prompt out for user to choose different algorithm.

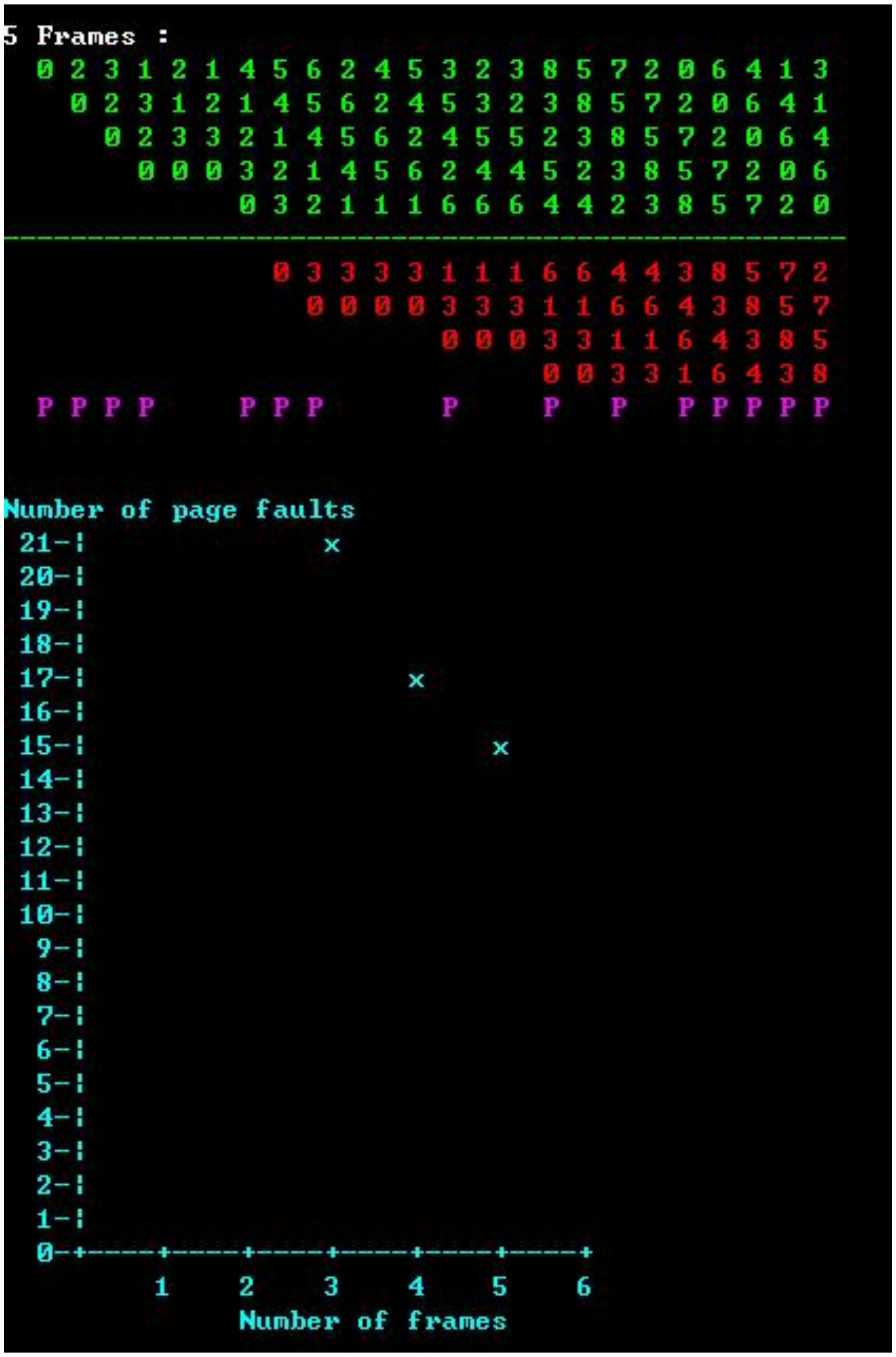


```
++++++++++++++++++++++++++++++
+ Paging Algorithm Simulation +
++++++++++++++++++++++++++++++

Choose a sample reference string :
1. 0231214562453238572064l3
2. 034324433430033
3. 01234567899876543210
========= OR ==========
4. Enter your own reference string.
9. Exit program.

Please enter a choice : 1


See the page replacement algorithm for
1. Least-Recently-Used (LRU)
2. Least-Frequently-Used (LFU)
3. First-In-First-Out (FIFO)
<< OTHER KEYS, return to main menu >>

Please enter a choice: _
```

After enter the choice for showing the algorithm, a sample output is as follow (For LRU).



```
Reference String : 0 2 3 1 2 1 4 5 6 2 4 5 3 2 3 8 5 7 2 0 6 4 1 3

Least-Recently-Used Algorithm (LRU)

(GREEN color represents pages that are in frame)
(RED color represents pages that have been swapped out)


3 Frames :
  0 2 3 1 2 1 4 5 6 2 4 5 3 2 3 8 5 7 2 0 6 4 1 3
    0 2 3 1 2 1 4 5 6 2 4 5 3 2 3 8 5 7 2 0 6 4 1
      0 2 3 3 2 1 4 5 6 2 4 5 5 2 3 8 5 7 2 0 6 4
--------------------------------------------------
        0 0 0 3 2 1 4 5 6 2 4 4 5 2 3 8 5 7 2 0 6
              0 3 2 1 4 5 6 2 2 4 5 2 3 8 5 7 2 0
                0 3 2 1 4 5 6 6 2 4 5 2 3 8 5 7 2
                  0 3 2 1 4 5 5 6 2 4 5 2 3 8 5 7
                    0 3 2 1 4 4 5 6 2 4 5 2 3 8 5
                      0 3 2 1 1 4 5 6 2 4 5 2 3 8
  P P P P     P P P P P P P P   P P P P P P P P P


4 Frames :
  0 2 3 1 2 1 4 5 6 2 4 5 3 2 3 8 5 7 2 0 6 4 1 3
    0 2 3 1 2 1 4 5 6 2 4 5 3 2 3 8 5 7 2 0 6 4 1
      0 2 3 3 2 1 4 5 6 2 4 5 5 2 3 8 5 7 2 0 6 4
        0 0 0 3 2 1 4 5 6 2 4 4 5 2 3 8 5 7 2 0 6
--------------------------------------------------
              0 3 2 1 1 1 6 6 6 4 4 2 3 8 5 7 2 0
                0 3 2 2 2 1 1 1 6 6 4 2 3 8 5 7 2
                  0 3 3 3 2 2 2 1 1 6 4 2 3 8 5 7
                    0 0 0 3 3 3 2 2 1 6 4 2 3 8 5
                          0 0 0 3 3 2 1 6 4 2 3 8
  P P P P     P P P P     P     P   P P P P P P P
```

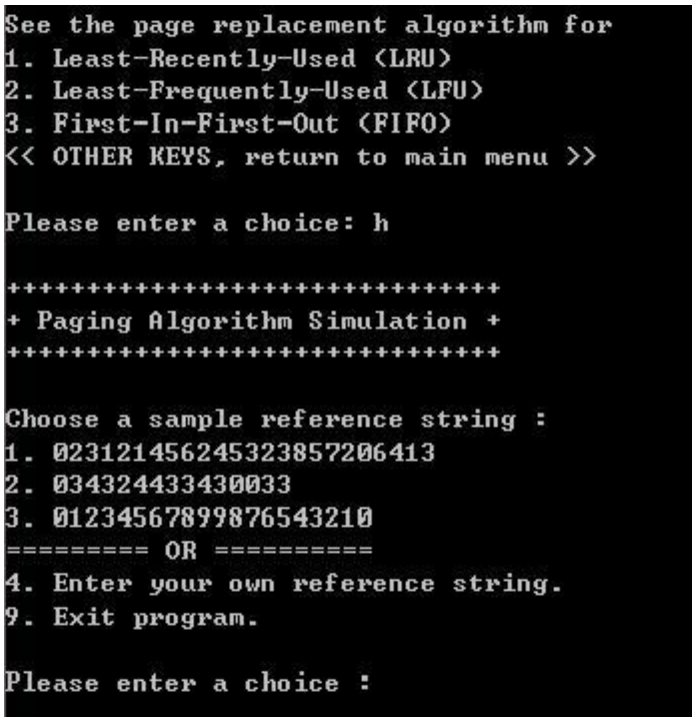


```
5 Frames :
  0 2 3 1 2 1 4 5 6 2 4 5 3 2 3 8 5 7 2 0 6 4 1 3
    0 2 3 1 2 1 4 5 6 2 4 5 3 2 3 8 5 7 2 0 6 4 1
      0 2 3 3 2 1 4 5 6 2 4 5 5 2 3 8 5 7 2 0 6 4
        0 0 0 3 2 1 4 5 6 2 4 4 5 2 3 8 5 7 2 0 6
              0 3 2 1 1 1 6 6 6 4 4 2 3 8 5 7 2 0
--------------------------------------------------
                0 3 3 3 3 1 1 1 6 6 4 4 3 8 5 7 2
                  0 0 0 0 3 3 3 1 1 6 6 4 3 8 5 7
                          0 0 0 3 3 1 1 6 4 3 8 5
                                0 0 3 3 1 6 4 3 8
  P P P P     P P P       P     P   P   P P P P P


Number of page faults
 21-¦              x
 20-¦
 19-¦
 18-¦
 17-¦                   x
 16-¦
 15-¦                        x
 14-¦
 13-¦
 12-¦
 11-¦
 10-¦
  9-¦
  8-¦
  7-¦
  6-¦
  5-¦
  4-¦
  3-¦
  2-¦
  1-¦
  0-+----+----+----+----+----+----+
         1    2    3    4    5    6
              Number of frames
```

Note: The page with green color means the page are in the frame.

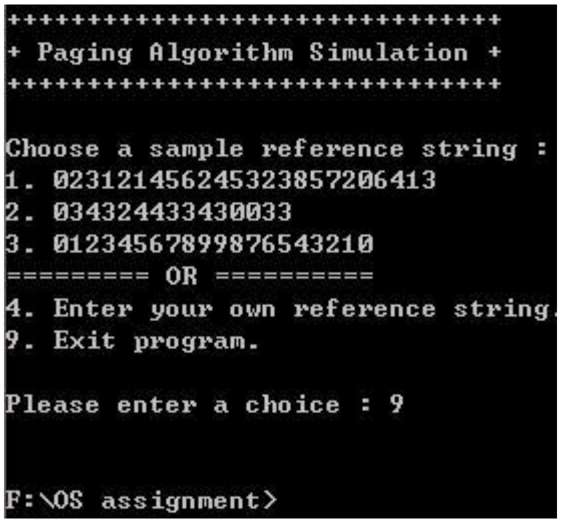
The page with red color means the page had been swapped out.

Enter others key other than 1, 2, and 3 to return to main menu.



```
See the page replacement algorithm for
1. Least-Recently-Used (LRU)
2. Least-Frequently-Used (LFU)
3. First-In-First-Out (FIFO)
<< OTHER KEYS, return to main menu >>

Please enter a choice: h

+++++++++++++++++++++++++++++++
+ Paging Algorithm Simulation +
+++++++++++++++++++++++++++++++

Choose a sample reference string :
1. 02312145624532385720641 3
2. 034324433430033
3. 01234567899876543210
========= OR =========
4. Enter your own reference string.
9. Exit program.

Please enter a choice :
```

Enter ‘9’ to exit the program.



```
+++++++++++++++++++++++++++++++
+ Paging Algorithm Simulation +
+++++++++++++++++++++++++++++++

Choose a sample reference string :
1. 023121456245323857206413
2. 034324433430033
3. 01234567899876543210
========= OR =========
4. Enter your own reference string.
9. Exit program.

Please enter a choice : 9


F:\OS assignment>
```