OS Lab7 Caching

1. Basic requirements (70)

   In this lab, you are required to write a simulator which can run FIFO, LRU, Min, Clock, Second-chance replacement algorithm. The explanations for these algorithms are as follows:

   1) FIFO: Using a FIFO list to maintain the element in cache. (**10**)
   2) Min: When page miss happens, you need to replace the one which we won't use for longest time. (**10**)
   3) LRU: When page miss happens, you need to replace the one which is least recent used. (**10**)
   4) Clock: We do a simplified version. Use a circular linked list to simulate the clock. The hand is always the last examined page. (The last query page) Every page in cache has a valid bit. When page miss happens, we start from the hand, and find the first page frame with valid bit 0. Replace this page and reset valid bit. Otherwise the hand increment. Before you move the hand, replace the valid bit of the current page to 0. (When you check, the hand does not move) (**15**)

      循环链表

   5) Second-chance: We do a simplified version. Divide the total cache-size into two parts with the same size. One uses FIFO, the other uses LRU. (**15**)
      a) We check in both FIFO list and LRU list.
      b) When the FIFO is not full, we just push in this part.
      c) When the FIFO is full and page miss happens, we move the last element in the FIFO list into the LRU list, and push the new element into the FIFO list.
      d) When the LRU list is full, we remove the least recent used element in the LRU list.
      e) When we find the query page in the LRU list, we remove it into the FIFO list.

   The allowed languages are C/C++. The compile command is: g++ -std=c++11.
   We will give you three test cases: 1.in(10000 pages), 2.in(100000 pages), 3.in(100000 pages)

Explanation for input test cases:

Please set your default cache size as 1024 unit.
The first line will be an integer N, which is the number of pages.
Then there will be N integers, represent the index of the query pages in order.

Please set the algorithm as the following format:

0-FIFO, 1-LRU, 2-Min, 3-Clock, 4-Second chance

Finally, your code will run on system test cases. (E.g. N = 1000000 and not generate randomly), if you can pass this test case using LRU efficiently; you can get the remaining **10** points. Our final input format is:

Cache size (an integer in [0 ... 2048])
Type of Algorithm {0, 1, 2, 3, 4}
Page size (10000000 >= N >= 100000 and not generate randomly)
Page1 Page2 … PageN

You should output the hit ratio as "Hit ratio = xx.xx%" (without quotes). Please remember, your result should be rounded up to 2 the decimal places. For example, 10.25% 9.99%.
If you use the given program, please modify the output. You are not allowed to print any other characters.

Your final score may be lower or higher based on your code quality.

2. Lab report(30)
   a) Write the report carefully and concretely. (10)
   b) We will give you three test cases. If you can complete the form correctly, you will get 20 points.