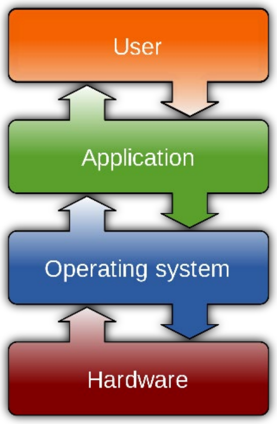
Operating Systems
Lab assignment 4: Memory Management



Objectives

1. To simulate a basic page replacement algorithm
2. To evaluate the performance, in terms of miss/hit rate, of this algorithm

Guidelines

The goal of this assignment is to gain experience with page replacement (and to a lesser extent, caching) algorithms. In this assignment you will write a program that simulates the FIFIO page replacement algorithm. Your program is to accept at least one numeric command-line parameter, which it will use as the number of available page frames. Additionally, your program will read in a txt file of page references.

These commands should run a simulation of the FIFO page replacement algorithm for a memory/cache size of cache_size pages/blocks. But from whence will page requests come? The answer is that your program should expect page requests to arrive on standard input (**stdin**, so a basic **fgets()**, or **scanf()** call should suffice to read in the unsigned integer page numbers being requested). So, assuming you have a sequence of page numbers in a text file called "accesses.txt" simulating the **FIFO** replacement algorithm you should be able to run your simulator by typing:

```
$cat accesses.txt | ./fifo cache_size
```

The output of your program will be every page number that was **not** found to be in the cache. In other words, the output of your program will be a sequence of page numbers that represents all the incoming requests that resulted in a page fault. Using your program, you should be able to get two numbers from the linux command line (by counting the number of lines read from the input file, and the number of lines produced by your simulator).The first of these numbers is the total number of page/block requests your simulator program has received (you get this by counting the number of valid lines in your input file), and the second number is how many of these page requests did result in a page fault (you get this by counting the number of lines produced as output by your program - which is faithfully reproducing the page replacement algorithm's behavior).

Your programs are to accept page requests on stdin as individual numbers, one per line, where each number indicates the requested page number. Each program is to further ignore any trailing text on the input lines, or any lines that do not start with a number. Your program terminates its simulation when it encounters an end-of-file. Once again, the size of the memory being managed by your program (the number of page frames, or the size of the cache if you treat this as a caching algorithm) is to be accepted as a command-line argument to your program. Any status output (e.g., messages you wish to print for debugging/user) should be sent to stderr (standard error, in other words, it should be possible to use your program and see nothing in standard output other than the page-faults/cache-misses, by redirecting only stdout).

You are to provide a program following replacement algorithm: FIFO.

Requirements to complete the lab

1. Show the TA your running page replacement simulator.
2. Write up a description of your implementations and sample miss-rate (page fault rate) results; submit it alongside your code. This portion of the assignment is as critical, if not more so, than the actual implementation of your solution. In addition to describing the code you provide a complete write-up will include a test of your solutions and a comparison of the hit rates for the different cache sizes you have implemented (plotting a graph would be strongly recommended). Your tests need include the provided accesses.txt test file provided. Test for memory sizes of between 10 and 1500 pages.