IE411 (OS), Win'24, Project 2: Page replacement simulator

In this project, you will simulate the virtual memory page replacement algorithms: Optimal (OPT), Least Recently Used (LRU) and First-In-First-Out (FIFO).

Due: 24th April, 23:50 hrs (via classroom)

You are to write three separate programs:

- a page replacement simulator vmsim,
- a page reference generator vmgen, and
- a page statistics program vmstats that produces a consolidated data file used for performance plots.

Section 1: Specifications of the page replacement simulator vmsim

- vmsim must accept three command-line arguments in the following order: (a) the total number of physical memory frames (maximum 100), (b) an input filename where a sequence of page references is stored, (c) the chosen algorithm. The three possible values for (c) are the strings opt, lru and fifo. (If vmsim is run with the wrong arguments or no arguments, it should print out usage instructions and exit.)

  Example:
  ```
  >vmsim 5 vmrefs.dat lru
  ```

- The format of the input file must be a simple ASCII sequence of integers in the range 0 to 99 separated by spaces, for example: 51 7 34 0 8 45 21. No symbols or formatted text, just a pure sequence of space-separated integer numbers. This file can be created by hand or generated by the vmgen program (see section 2 below).

- vmsim will first read all the memory references from the input file and store them in a local array. Then, it will play back these references one by one and print out for each reference the current allocation state of physical memory frames in the following format:

  ```
  34: [51| 7|34|  |  ]
  ```

- This line means that after using page 34, frames 0, 1 and 2 are occupied by pages 51, 7 and 34, and frames 3 and 4 are empty. Frames must start with open square bracket [, end with closed square bracket] and be separated with vertical bar |. One-digit page numbers should have an extra space to the left so that frames are always 2 characters wide (2 spaces for an empty frame). Each page fault should be signaled by an F character two spaces to the right of the closed bracket, for example:

  ```
  45: [45| 7|34| 0| 8]  F
  ```

- After processing all the memory references, vmsim should finally print the total number of page faults and the miss rate (page faults divided by number of references). It should start counting page faults and page references only after all frames have been initially filled, e.g., with 3 frames that means starting at the 4th step. Use this printout format:

  ```
  Miss rate = 237 / 997 = 23.78%
  ```

Section 2: Specifications of the page reference generator vmgen

- vmgen must accept three command-line arguments in the following order: (a) the range of page references (maximum 100), (b) the length of the sequence and (c), the name of the file that will be generated. (If vmgen is run with the wrong arguments or no arguments, it should print out usage instructions and exit.) Example:

  ```
  >vmgen 10 200 vmrefs.dat
  ```

- vmgen will then generate a sequence of the desired length containing random page numbers uniformly drawn between 0 and the range minus one (i.e., 200 page numbers between 0 and 9 in the example above). Important: no page number in the sequence should be equal to the number that precedes it (hence, follows it, too). For example: 2 7 7 0 0 0 3 3 … is not a valid sequence but 2 7 0 3 … is. vmgen must write this sequence into the file given in input.

Section 3: Specifications of the page statistics program vmstats

- vmstats must accept four command-line arguments in the following order: (a) the minimum number of frames (no less than 2), (b) the maximum number of frames (no more than 100), (c) the frame number increment (positive), (d) the input filename containing the references. (If vmstats is run with the wrong arguments or no arguments, it should print out usage instructions and exit.) Example:

  ```
  >vmstats 5 40 10 vmrefs.dat
  ```

- vmstats will loop over the three page replacement methods: opt, lru and fifo and for each method, it will loop over the number of frames, starting at the minimum and applying the increment until it exceeds the maximum (i.e., 5, 15, 25, 35, in the example above). For each method/number of frames pair it will calculate the page fault rate using the reference file given in input and print out a one-line message containing this rate. Use this printout format:

  ```
  lru, 15 frames: Miss rate = 237 / 985 = 23.78%
  lru, 25 frames: Miss rate = 163 / 975 = 16.35%
  ...
  ```

- Therefore, the implementation of both vmsim and vmstats must rely on a common utility simulation engine (entry-point function) with the difference that vmsim is going to run the simulation once and in verbose mode (displaying all the allocation steps and the final miss rate; see Section 1), whereas vmstats is going to run the simulation repeatedly and silently (displaying only the final miss rate for each pair in the loops). In accordance with this design, please keep the simulation engine in a source file distinct from the vmsim and vmstats code.

- As a by-product, vmstats must also generate one consolidated results file containing the matrix of all the calculated miss rates. The results file must be named vmrates.dat and must be formatted in the following way: it should contain exactly four lines of N space-separated numbers, where N is the number of numbers of frames (4 in the example above). The first line must be the sequence of numbers of frames (5 15 25 35 in the example above), the other three lines must be the sequences of rate values for each number of frame. These sequences must be in the following order: opt, lru and fifo. vmrates.dat should not contain formatted messages or symbols, just four lines of pure space-separated numerical sequences with line feeds between sequences.