CISC 3320 HW Assignment – 4 (12 pts), EC (7 pts)

This assignment has two parts, and both involve memory management. There is also an extra credit part-3 which is an extension of part-2 and carries an extra 7-point. You can use either C/C++ or Java for your development.

Requirements

Part-1 (3 pts) - Write a program named: `TranslateAddr`

1. The program should assume a 32-bit virtual (logical) address with a 4-KB page size, and an address in decimal will be passed in via the command line
2. The program verifies that the address is within the 32-bit range (exits and displays an error message if the input is out of the range)
3. It then calculates and prints to screen the following (assuming a one-level page table with 4 bytes for each entry of the table, and the command-line input is 20221108):
   ```
   The largest possible page number is: ????
   The page table size is: ???? bytes
   Given the address of 20221108
   The page number is: ????
   The page offset is: ????
   ```
4. Additionally, assume a two-level page table (10-bit each), calculate and print the following:
   ```
   With a 2-level page table, the outer page number (p1) is: ????
   With a 2-level page table, the inner page number (p2) is: ????
   ```
   - Refer to slides 9.40-41 in Ch9 for the meaning of p1 and p2.

Part-2 (9 pts) - Write a program named: `ReplacePage`

1. In separate methods, implement the FIFO and LRU page-replacement algorithms as presented in Ch10 (slides 10.35 and 10.38, respectively).
   - Allow each method to accept as arguments: a reference string, a desired number of page frame, and optionally a flag to enable/disable output (used for part-3 extra credit)
   - Each method should return the number of page fault
2. As a test case, use 3 as number of page frame and the exact reference string shown on the slides:
   ```
   String testRefStr = "70120304230321201701"; // in Java
   ```
   - Print out results for both FIFO and LRU. A sample example for FIFO is shown below:
   ```
   FIFO: for ref string: 70120304230321201701, with 3 frames
   7: 7
   0: 70
   1: 701
   2: 201
   0:
   3: 231
   0: 230
   4: 430
   2: 420
   3: 423
   0: 023
   3:
   2:
   ```

```
1: 013
2: 012
0:
1:
7: 712
0: 702
1: 701
FIFO: total # of page fault is: 15
```

3. Additionally, for FIFO, show the effect of Belady's Anomaly by using this reference string:
   `String testRefStr2 = "123412512345"; // in Java`
   Print out results for 3 and 4 page frames – they should match the result presented on slide 10.35

Part-3 Extra Credit (7 pts) - Modify your `ReplacePage` to create a third program, `ReplacePageEC`

1. Add a new method that implements the OPT page-replacement algorithm (slide 10.37)
2. Notice that this algorithm looks into the future, thus is impossible to use. Rather it is implemented as a benchmark to evaluate other algorithms which you will do in this exercise
3. First make sure it's implemented correctly by employing the same `testRefStr` as shown above and display the resultant number of page fault
4. Then, your program should do a simple comparison analysis of FIFO and LRU against the benchmark of OPT
   - Write another method `randomPageRef()` which takes in as input an integer for length and returns a randomized reference string of that length using characters '0' - '9'
   - Generate a reference string that is 1000 characters long
   - Try this reference string on OPT, FIFO and LRU with number of page frame ranging from 1 to 7, and print out the number of page fault as follows:

     # of page frame: # of page fault for OPT, for FIFO (% more than OPT), for LRU (% more than OPT)

     Sample output (partial)
     ```
     1: 915, 915(0%), 915(0%)
     2: 654, 812(24%), 816(25%)
     ```

What to turn in

- Please submit (via email) the following:
  - your source code in separate files
  - running results (you could use a screen capture, but preferably a text file that captures the result)
  - you can capture the result by redirecting the output to a file:
    - in C/C++: `$./ReplacePage > result.txt`
    - in Java: `$java ReplacePage > result.txt`
- **Important**: please make sure your submission has all required results. Specifically, it should have:
  - `testRefStr` results for FIFO and LRU with 3 frames
  - `testRefStr2` results for FIFO with 3 and 4 frames
  - For extra credit, `testRefStr` result for OPT with 3 frames, followed by a list of comparison results
  - If you submit partial result, make it clear in your description
- Please do not copy code, whether from classmates or from the Internet.