CSE 5/7343 – Program 5: Enhanced Clock Page Replacement

Submission due Friday, April 20 by 11:59PM

You are to write a c program that implements the enhanced clock page replacement algorithm as explained on pp 368-9. The clock will be implemented as a 4-element circular array. Write your code in such a way that it is easy to change the clock size. Each clock entry will contain a use bit, a modify bit and a logical page number.

The program will be tested using data text file testdata.txt. Each "record" of the file contains two fields: a page number and read/write character that indicates if the page is to be read from or written to.

testdata.txt

4r
3w
6w
1r
5r
5w
2w
5r
8w
9w
0r
5r
2r
7w
4w

The clock algorithm performs as follows:

1. Read a record.
2. Check to see if the page in the record is already in the clock and if it is update to appropriate flag(s). (This will not reposition the next frame pointer)
3. If step 2 fails, beginning at the current position of the next frame pointer scan the clock. During this scan make no changes to the use bit. The first frame encountered with (u = 0; m = 0) is selected for replacement.
4. If step 3 fails, scan the clock again. The first frame encountered with (u = 0; m = 1) is selected for replacement. During this scan, set the use bit to 0 for each frame that is bypassed.
5. If step 4 fails, the frame pointer will have once again returned to its original position and all the frames will have a use bit of 0. Repeat step 3 and, if necessary, step 4. By time a frame will be found for replacement.
6. Write the updated clock to file results.txt.

7. Repeat 1-6 until end of testdata.txt reached

Since file processing in c will be new to many of you I have provided some code to help you get started.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#define CLOCK_SIZE 4

void writeClockToFile(FILE* filePtr, additional parameters) {
    int i = 0;
    fprintf(filePtr, " FRAME       PAGE       USE       MODIFY\n");
    for (  ; i < CLOCK_SIZE; i++) {


    }
};

int main()  {

    char inFileName[ ] = "testdata.txt";
    FILE *inFilePtr = fopen(inFileName, "r");
    if(inFilePtr == NULL) {
        printf("File %s could not be opened.\n", inFileName);
        exit(1);
    }

    char outFileName[ ] = "results.txt";
    FILE *outFilePtr = fopen(outFileName, "w");
    if(outFilePtr == NULL) {
        printf("File %s could not be opened.\n", outFileName);
        exit(1);
    }

    fscanf(inFilePtr, "%d%c", &page, &operation);
    while(!feof(inFilePtr)) {
        fprintf(outFilePtr, "Page referenced: %d %c\n", page, operation);
```

```
        writeClockToFile(outFilePtr, additional arguments);
        fscanf(inFilePtr, "%d%c", &page, &operation);
    }//end while

    fclose(inFilePtr);
    fclose(outFilePtr);
    return 0;
};
```

After each page reference has been processed, write your clock to *results.txt* using the following format:

results.txt

```
Page referenced: 4 r
FRAME  PAGE  USE   MODIFY
  0      4     1      0
  1     -1     0      0  <- next frame
  2     -1     0      0
  3     -1     0      0

Page referenced: 3 w
FRAME  PAGE  USE   MODIFY
  0      4     1      0
  1      3     1      1
  2     -1     0      0  <- next frame
  3     -1     0      0

Page referenced: 6 w
FRAME  PAGE  USE   MODIFY
  0      4     1      0
  1      3     1      1
  2      6     1      1
  3     -1     0      0  <- next frame

Page referenced: 1 r
FRAME  PAGE  USE   MODIFY
  0      4     1      0  <- next frame
  1      3     1      1
  2      6     1      1
  3      1     1      0

Page referenced: 5 r
FRAME  PAGE  USE   MODIFY
  0      5     1      0
```

1	3	0	1	<- next frame
2	6	0	1	
3	1	0	0	

Page referenced: 5 w

FRAME	PAGE	USE	MODIFY	
0	5	1	1	
1	3	0	1	<- next frame
2	6	0	1	
3	1	0	0	

Page referenced: 2 w

FRAME	PAGE	USE	MODIFY	
0	5	1	1	<- next frame
1	3	0	1	
2	6	0	1	
3	2	1	1	

Page referenced: 5 r

FRAME	PAGE	USE	MODIFY	
0	5	1	1	<- next frame
1	3	0	1	
2	6	0	1	
3	2	1	1	

Page referenced: 8 w

FRAME	PAGE	USE	MODIFY	
0	5	0	1	
1	8	1	1	
2	6	0	1	<- next frame
3	2	1	1	

Page referenced: 9 w

FRAME	PAGE	USE	MODIFY	
0	5	0	1	
1	8	1	1	
2	9	1	1	
3	2	1	1	<- next frame

Page referenced: 0 r

FRAME	PAGE	USE	MODIFY	
0	0	1	0	
1	8	1	1	<- next frame
2	9	1	1	

3	2	0	1

Page referenced: 5 r

FRAME	PAGE	USE	MODIFY	
0	0	1	0	<- next frame
1	8	0	1	
2	9	0	1	
3	5	1	0	

Page referenced: 2 r

FRAME	PAGE	USE	MODIFY	
0	0	0	0	
1	2	1	0	
2	9	0	1	<- next frame
3	5	1	0	

Page referenced: 7 w

FRAME	PAGE	USE	MODIFY	
0	7	1	1	
1	2	1	0	<- next frame
2	9	0	1	
3	5	1	0	

Page referenced: 4 w

FRAME	PAGE	USE	MODIFY	
0	7	1	1	
1	2	0	0	
2	4	1	1	
3	5	1	0	<- next frame