Project 3 - Paging

Project Overview and Coding Tasks

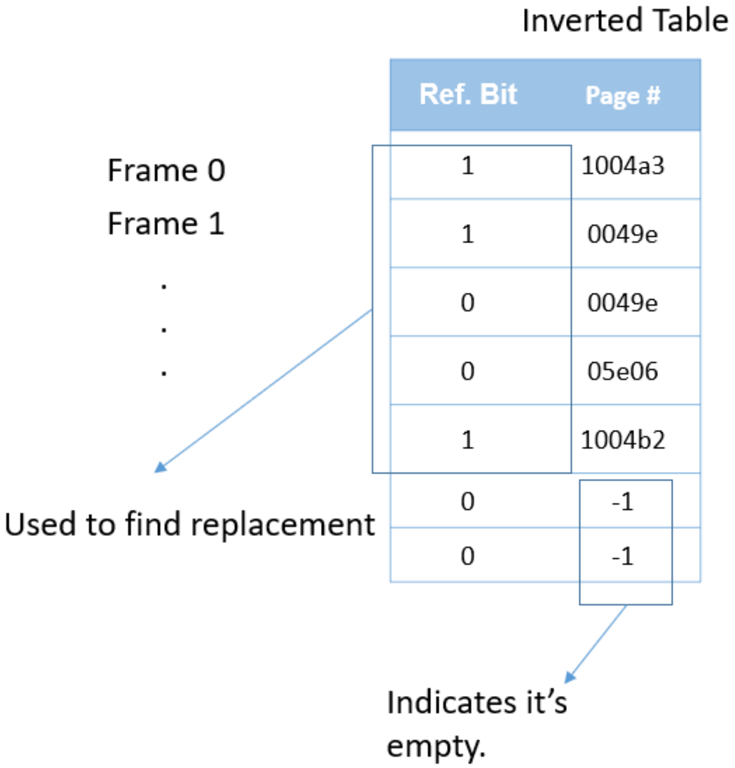
In this programming assignment, you will implement a simulation of Second Chance Paging for different page sizes while using an inverted page table.

You will need to calculate the number of page faults for different combinations of pages sizes and memory sizes.

Page sizes: 4KB, 16KB, 32KB, and 64KB
Memory sizes: 64KB, 128KB, and 256KB

Inverted Page Table

The table will contain the page addresses of occupying pages and the reference bits. Your table will not include a ASID as we are only running one process in this project assignment. The paging system utilizes 32-bit.

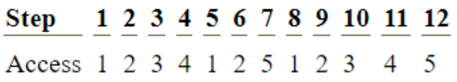


Ref. Bit	Page #
1	1004a3
1	0049e
0	0049e
0	05e06
1	1004b2
0	-1
0	-1

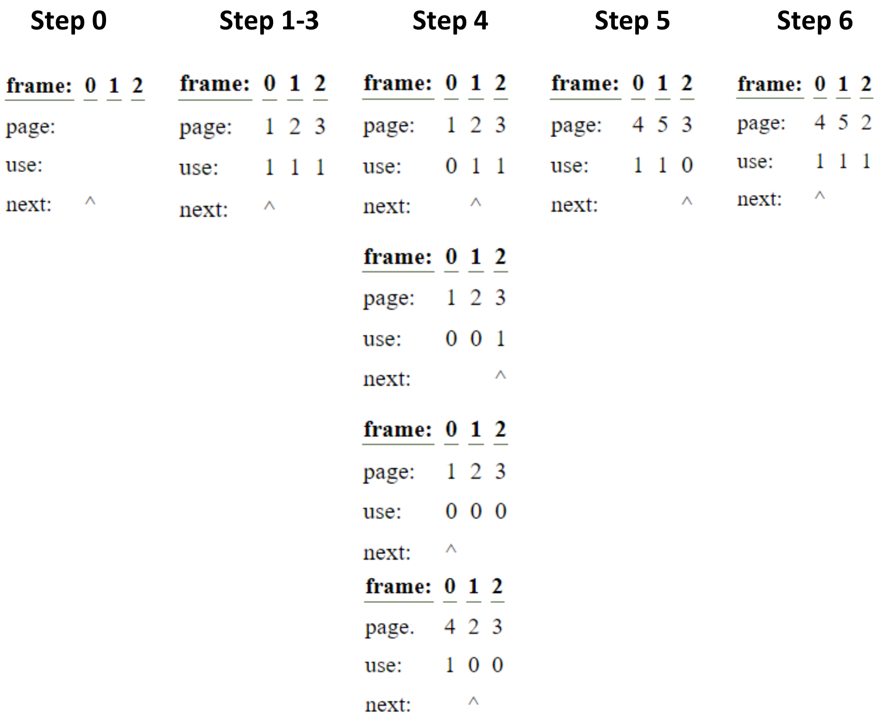
Second Chance Page Replacement

Reference bits are upated each time the frame is accessed, newly fetched, or when there is a page fault. Refer to the example below to know when the pointer “^” is moving to the next address and when the reference bit “use” is being updated.

Example:



Step	1	2	3	4	5	6	7	8	9	10	11	12
Access	1	2	3	4	1	2	5	1	2	3	4	5



Step 0

frame:	0	1	2
page:			
use:			
next:	^		

Step 1-3

frame:	0	1	2
page:	1	2	3
use:	1	1	1
next:	^		

Step 4

frame:	0	1	2
page:	1	2	3
use:	0	1	1
next:		^	

frame:	0	1	2
page:	1	2	3
use:	0	0	1
next:			^

frame:	0	1	2
page:	1	2	3
use:	0	0	0
next:	^		

frame:	0	1	2
page.	4	2	3
use:	1	0	0
next:		^	

Step 5

frame:	0	1	2
page:	4	5	3
use:	1	1	0
next:			^

Step 6

frame:	0	1	2
page:	4	5	2
use:	1	1	1
next:	^		

Input Files

Your program must prompt the user to input the name of the input file, the page size and the mememory size. The input file with be formatted as follows:

0052h8a0 R

0052h8a0 W

..

Each line contains eight characters that are a representation of a 32-bit hexadecimal address, followed by a letter ‘R’ or ‘W’ indicating a read or write. Taking into account the page size used, you program will parse the page number of each address. You will ignore the read and write characters as they have no use in this simulation.

Command line Arguments

Your main function “**Simulation”** (in **Simulation.c** or **Simulation.java)** should prompt the user to input arguments in the following order:

1. Name of the file used.
2. An integer value indicating memory size in KB
3. An integer value indicating page size in KB

As an example: trace1.txt 64 4

A sample output would look like this:

```
--------------------------------------------

Trace File: trace1.txt

Memory Size: 64 KB

Page Size: 4 KB

Page Faults: 191849

--------------------------------------------
```

Files to submit

A zipped file containing the entirety of the implemented code with all source files included.

A 1 page report, that includes your results in a table for borth trace files, and a concluding analysis on the best performing parameter values for each trace file.

You will be graded on the following criteria:

Code readibilty	10%
Correctness	55%
Code is compilable	15%
Your report’s brief discussion and explanation of results	20%