Problem Statement 13 Optimal Page Replacement

import java.util.ArrayList;

import java.util.HashMap;

public class OptimalPageReplacement {

static int pageFrames = 3;

static int optimal(int referenceString[]) {

ArrayList<Integer> pages = new ArrayList<>(pageFrames);

HashMap<Integer, Integer> indexes = new HashMap<>();

int page\_faults = 0, curPage, n = referenceString.length;

for (int i = 0; i < n; i++) {

curPage = referenceString[i];

if (pages.size() < pageFrames) {

if (!pages.contains(curPage)) {

pages.add(curPage);

page\_faults++;

displayPageFrames(pages, page\_faults);

}

indexes.put(curPage, findNextIndex(curPage, i, referenceString));

} else {

if (!pages.contains(curPage)) {

int optimal = Integer.MIN\_VALUE, pageToBeReplaced = 0;

for (int j = 0; j < pages.size(); j++) {

int temp = pages.get(j);

if (indexes.get(temp) > optimal) {

optimal = indexes.get(temp);

pageToBeReplaced = j;

}

}

indexes.remove(pages.get(pageToBeReplaced));

pages.set(pageToBeReplaced, curPage);

page\_faults++;

displayPageFrames(pages, page\_faults);

}

indexes.put(curPage, findNextIndex(curPage, i, referenceString));

}

}

return page\_faults;

}

static int findNextIndex(int curPage, int curIndex, int pages[]) {

for (int i = curIndex + 1; i < pages.length; i++) {

if (pages[i] == curPage) {

return i;

}

}

return Integer.MAX\_VALUE;

}

static void displayPageFrames(ArrayList<Integer> pages, int page\_faults) {

System.out.print("At PageFault- " + page\_faults + " :: Pages- ");

for (int page : pages) {

System.out.print(" " + page);

}

System.out.println();

}

public static void main(String args[]) {

System.out.println("Optimal Page Replacement Algorithm");

int pages[] = {7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1};

int pageFaults = optimal(pages);

System.out.println("Total page faults = " + pageFaults);

}

}