LRU Program

import java.util.\*;  
class LRU  
{  
 public static void main(String args[])  
 {  
 Scanner sc=new Scanner(System.in);  
  
 System.out.println("ENTER NUMBER OF FRAMES");  
 int nf=sc.nextInt();  
 System.out.println("ENTER NUMBER OF REFERENCES");  
 int nr=sc.nextInt();  
 int page[]=new int[nr];  
 int frame[]=new int[nr];  
                System.out.println("ENTER REFERENCE");  
 for(int i=0;i<nr;i++)  
 page[i]=sc.nextInt();  
 for(int i=0;i<nf;i++)  
 frame[i]=-1;  
 for(int k=0;k<nf;k++)  
 System.out.print(" "+frame[k]);  
 System.out.println();  
 int flag=0,hit=0,miss=0,front=0,rear=-1;  
 int age[]=new int[nf];  
 for(int i=0;i<nf;i++)  
 age[i]=0;  
 for(int j=0,i=0;j<nf && i<nr;j=j%nf,i++)  
 {  
 flag=0;  
   
 for(int y=0;y<nf;y++)  
 {  
 if(frame[y]==page[i])  
 {  
 flag=1;  
 hit++;  
 age[y]=0;   
 }  
 }  
 if(flag==0)  
 if(frame[j]==-1)  
 {   
   
 frame[j]=page[i];  
                                        miss++;  
 rear++;  
 j++;  
 }  
 else  
   
 {  
 int max=age[0],loc=0;  
 miss++;  
  
  
 for(int b=0;b<nf;b++)  
 if(age[b]>max)  
 {  
 max=age[b];loc=b;  
 }  
 frame[loc]=page[i];  
 age[loc]=0;  
 j++;  
                                        }   
 for(int k=0;k<nf;k++)  
 System.out.print(" "+frame[k]);  
 System.out.println();  
 for(int a=0;a<nf;a++)  
 {  
 if(frame[a]==-1)  
 age[a]=0;  
 else  
 age[a]++;  
 }  
   
 /\*for(int k=0;k<nf;k++)  
 System.out.print(" "+age[k]);  
 System.out.println();\*/  
 }  
 float hr=(float)hit/((float)hit+(float)miss);  
 System.out.println("Hit="+hit+" miss="+miss);  
 System.out.println("page replacement ratio="+hr);  
 }  
}

optimal

|  |
| --- |
| import java.util.Scanner; |
|  | import java.io.IOException; |
|  |  |
|  | public class OptimalPageReplacement |
|  | { |
|  | public static void main(String[] args) throws IOException |
|  | { |
|  | Scanner in = new Scanner(System.in); |
|  | int frames = 0; |
|  | int pointer = 0; |
|  | int numFault = 0; |
|  | int ref\_len; |
|  | boolean isFull = false; |
|  | int buffer[]; |
|  | boolean hit[]; |
|  | int fault[]; |
|  | int reference[]; |
|  | int mem\_layout[][]; |
|  |  |
|  | System.out.println("Please enter the number of frames: "); |
|  | frames = Integer.parseInt(in.nextLine()); |
|  |  |
|  | System.out.println("Please enter the length of the reference string: "); |
|  | ref\_len = Integer.parseInt(in.nextLine()); |
|  |  |
|  | reference = new int[ref\_len]; |
|  | mem\_layout = new int[ref\_len][frames]; |
|  | buffer = new int[frames]; |
|  | hit = new boolean[ref\_len]; |
|  | fault = new int[ref\_len]; |
|  | for(int j = 0; j < frames; j++) |
|  | { |
|  | buffer[j] = -1; |
|  | } |
|  |  |
|  | System.out.println("Please enter the reference string (hit Enter/Return after each number in the string): "); |
|  | for(int i = 0; i < ref\_len; i++) |
|  | { |
|  | reference[i] = Integer.parseInt(in.nextLine()); |
|  | } |
|  | System.out.println(); |
|  | for(int i = 0; i < ref\_len; i++) |
|  | { |
|  | int search = -1; |
|  | for(int j = 0; j < frames; j++) |
|  | { |
|  | if(buffer[j] == reference[i]) |
|  | { |
|  | search = j; |
|  | hit[i] = true; |
|  | fault[i] = numFault; |
|  | break; |
|  | } |
|  | } |
|  |  |
|  | if(search == -1) |
|  | { |
|  | if(isFull) |
|  | { |
|  | int index[] = new int[frames]; |
|  | boolean index\_flag[] = new boolean[frames]; |
|  | for(int j = i + 1; j < ref\_len; j++) |
|  | { |
|  | for(int k = 0; k < frames; k++) |
|  | { |
|  | if((reference[j] == buffer[k]) && (index\_flag[k] == false)) |
|  | { |
|  | index[k] = j; |
|  | index\_flag[k] = true; |
|  | break; |
|  | } |
|  | } |
|  | } |
|  | int max = index[0]; |
|  | pointer = 0; |
|  | if(max == 0) |
|  | { |
|  | max = 200; |
|  | } |
|  |  |
|  | for(int j = 0; j < frames; j++) |
|  | { |
|  | if(index[j] == 0) |
|  | { |
|  | index[j] = 200; |
|  | } |
|  |  |
|  | if(index[j] > max) |
|  | { |
|  | max = index[j]; |
|  | pointer = j; |
|  | } |
|  | } |
|  | } |
|  | buffer[pointer] = reference[i]; |
|  | numFault++; |
|  | fault[i] = numFault; |
|  | if(!isFull) |
|  | { |
|  | pointer++; |
|  | if(pointer == frames) |
|  | { |
|  | pointer = 0; |
|  | isFull = true; |
|  | } |
|  | } |
|  | } |
|  |  |
|  | for(int j = 0; j < frames; j++) |
|  | { |
|  | mem\_layout[i][j] = buffer[j]; |
|  | } |
|  | } |
|  |  |
|  | for(int i = 0; i < ref\_len; i++) |
|  | { |
|  | System.out.print(reference[i] + ": Memory is: "); |
|  | for(int j = 0; j < frames; j++) |
|  | { |
|  | if (mem\_layout[i][j] == -1) |
|  | { |
|  | System.out.printf("%3s ", "\*"); |
|  | } else |
|  | { |
|  | System.out.printf("%3d ", mem\_layout[i][j]); |
|  | } |
|  | } |
|  | System.out.print(": "); |
|  | if (hit[i]) { |
|  | System.out.print("Hit"); |
|  | } else |
|  | { |
|  | System.out.print("Page Fault"); |
|  | } |
|  | System.out.print(": (Number of Page Faults: " + fault[i] + ")"); |
|  | System.out.println(); |
|  | } |
|  | System.out.println("Total Number of Page Faults: " + numFault); |
|  | } |
|  | } |