MCA I SEMESTER -108p:OSLAB

1. Write a java program to implement the concept of first in first out job scheduling.

2. Write a java program to implement the concept of shortest job first scheduling.

3. Write a java program to implement the concept of round robin job scheduling.

4. Write a java program to implement the concept of priority scheduling.

5. Write a java program to implement the concept of bankers algorithm.

6. Write a java program to implement the concept of dining philosopher problem

7. Write a java program to implement the concept of producer and consumer problem

8. Write a java program to implement best fit algorithm

9. Write a java program to implement the concept of memory fragmentation.

10. Write a java program to implement the concept of FIFO page replacement.

11. Write a java program to implement the concept of LRU page replacement.

12. Write a java program to implement the concept of optimal page replacement.

13.Write a java program to implement the concept of SCAN disk scheduling.

14. Write a java program to implement the concept of CLOOK disk scheduling.

15. Write a java program to implement the concept of FCFS disk scheduling.

```
//1. FCFS SCHEDULING
import java.util.*;
class FCF
{
public static void main(String args[])
{
Scanner in=new Scanner(System.in);
int p;
float t1=0,t2=0;
System.out.println("Enter number of processes");
p=in.nextInt();
int bt[]=new int[p];

System.out.println("Enter burst time");
for(int i=0;i<p;i++)
{
System.out.println("Burst time for P"+(i+1)+"=");
bt[i]=in.nextInt();
}
int wt[]=new int[p];
wt[0]=0;
for(int i=1;i<p;i++)
{
wt[i]=bt[i-1]+wt[i-1];
t1+=wt[i];
}
int tat[]=new int[p];
for(int i=0;i<p;i++)
{
tat[i]=bt[i]+wt[i];
t2+=tat[i];
}
System.out.println("Process\tBurst time\tWaiting time\tTurn Around time");
for(int i=0;i<p;i++)
{
System.out.println("P"+(i+1)+"\t\t"+bt[i]+"\t\t"+wt[i]+"\t\t"+tat[i]);
}
float awt,atat;
System.out.println("Average Waiting time="+t1/p);
System.out.println("Average Turn Around time="+t2/p);
System.out.println("_____");
System.out.println("GANT CHART");
System.out.println("_____");
for(int i=1;i<=p;i++)
{System.out.print("P"+i);
System.out.print("\t->");}
System.out.println("\n_____");
System.out.print("0");
for(int i=0;i<p;i++)
{
System.out.print("\t"+(wt[i]+bt[i]));}
```

```
}
}
/*OUTPUT
D:\Java>java FCF
Enter number of processes:3
Enter burst time
Burst time for P1=24
Burst time for P2=3
Burst time for P3=3
Process Burst time     Waiting time    Turn Around time
P1          24          0          24
P2          3           24         27
P3          3           27         30
Average Waiting time=17.0
Average Turn Around time=27.0
_____________________________
GANTT CHART
_____________________________
P1     ->P2   ->P3   ->
_____________________________
0      24     27     30
*/
```

```
//2. SHORTEST JOB FIRST SCHEDULING(SJF)
import java.util.*;
class SJF
{
public static void main(String args[])
{
 Scanner sc=new Scanner(System.in);
int n,BT[],WT[],TAT[];
System.out.println("Enter no of process");
n=sc.nextInt();
BT=new int[n+1];
WT=new int[n+1];
TAT=new int[n+1];
float AWT=0;
System.out.println("Enter Burst time for each process");
for(int i=0;i<n;i++)
{
System.out.println("Enter BT for process "+(i+1));
BT[i]=sc.nextInt();
}
for(int i=0;i<n;i++)
{WT[i]=0;
 TAT[i]=0;
}
int temp;
for(int i=0;i<n;i++)
{for(int j=0;j<n-1;j++)
{ if(BT[j]>BT[j+1])
  {temp=BT[j];
BT[j]=BT[j+1];
BT[j+1]=temp;
temp=WT[j];
WT[j]=WT[j+1];
WT[j+1]=temp;
}
}
}
for(int i=0;i<n;i++)
{TAT[i]=BT[i]+WT[i];
WT[i+1]=TAT[i];
}
TAT[n]=WT[n]+BT[n];
System.out.println(" PROCESS  BT  WT  TAT   ");
for(int i=0;i<n;i++)
System.out.println("   "+ i + "     "+BT[i]+"     "+WT[i]+"     "+TAT[i]);
for(int j=0;j<n;j++)
AWT+=WT[j];
AWT=AWT/n;
System.out.println("************************************************");
System.out.println("Avg waiting
time="+AWT+"\n*************************************************");
```

```
}
}


/*OUTPUT
Enter no of process
4
Enter Burst time for each process
Enter BT for process 1
5
Enter BT for process 2
3
Enter BT for process 3
3
Enter BT for process 4
1
 PROCESS   BT     WT     TAT
   0         1       0       1
   1         3       1       4
   2         3       4       7
   3         5       7      12
***********************************************
Avg waiting time=3.0
***********************************************
```

```
//3.ROUND ROBIN SCHEDULING
import java.io.*;
class round
{
public static void main(String args[])throws IOException
{
DataInputStream in=new DataInputStream(System.in);
int i,j,k,q,sum=0;
System.out.println("Enter number of process:");
int n=Integer.parseInt(in.readLine());
int bt[]=new int[n];
int wt[]=new int[n];
int tat[]=new int[n];
int a[]=new int[n];
System.out.println("Enter brust Time:");
for(i=0;i<n;i++)
{
System.out.println("Enter brust Time for   "+(i+1));
bt[i]=Integer.parseInt(in.readLine());
}
System.out.println("Enter Time quantum:");
q=Integer.parseInt(in.readLine());
for(i=0;i<n;i++)
a[i]=bt[i];
for(i=0;i<n;i++)
wt[i]=0;
do
{
for(i=0;i<n;i++)
{
if(bt[i]>q)
{
bt[i]-=q;
for(j=0;j<n;j++)
{
if((j!=i)&&(bt[j]!=0))
wt[j]+=q;
}}
else
{
for(j=0;j<n;j++)
{
if((j!=i)&&(bt[j]!=0))
wt[j]+=bt[i];
}
bt[i]=0;
}}
sum=0;
for(k=0;k<n;k++)
sum=sum+bt[k];
}
```

```
while(sum!=0);
for(i=0;i<n;i++)
tat[i]=wt[i]+a[i];
System.out.println("process\tBT\tWT\tTAT");
for(i=0;i<n;i++)
{
System.out.println("process"+(i+1)+"\t"+a[i]+"\t"+wt[i]+"\t"+tat[i]);
}
float avg_wt=0;
float avg_tat=0;
for(j=0;j<n;j++)
{
avg_wt+=wt[j];
}
for(j=0;j<n;j++)
{
avg_tat+=tat[j];
}
System.out.println("average wating time "+(avg_wt/n)+"\nAverage turn around
time"+(avg_tat/n));
}}
```

OUTPUT

```
/*Round robin Scheduling algorithm output:
Enter number of process:4
Enter brust Time:
Enter brust Time for  1:4
Enter brust Time for  2:5
Enter brust Time for  3:6
Enter brust Time for  4:7
Enter Time quantum:4
process       BT     WT     TAT
process1      4      0       4
process2      5      12      17
process3      6      13      19
process4      7      15      22
average wating time    10.0
Average turn around time15.5
```

```
// 4. PRIORITY SCHEDULING ALGORITHM:
import java.util.Scanner;
class PrioritySchedulingOS {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter number of processes: ");
        int n = sc.nextInt();

        int pid[] = new int[n];
        int bt[] = new int[n];
        int pr[] = new int[n];
        int wt[] = new int[n];
        int tat[] = new int[n];
        // Input process details
        for (int i = 0; i < n; i++) {
            pid[i] = i + 1;
            System.out.print("Enter Burst Time for P" + pid[i] + ": ");
            bt[i] = sc.nextInt();
            System.out.print("Enter Priority for P" + pid[i] + ": ");
            pr[i] = sc.nextInt();
        }

        // Sort by priority (OS scheduler behavior)
        for (int i = 0; i < n - 1; i++) {
            for (int j = i + 1; j < n; j++) {
                if (pr[i] > pr[j]) {
                    int temp = pr[i];
                    pr[i] = pr[j];
                    pr[j] = temp;
```

```
                temp = bt[i];
                bt[i] = bt[j];
                bt[j] = temp;

                temp = pid[i];
                pid[i] = pid[j];
                pid[j] = temp;
            }
        }
    }

    // Waiting Time calculation
    wt[0] = 0;
    for (int i = 1; i < n; i++) {
        wt[i] = wt[i - 1] + bt[i - 1];
    }
    // Turnaround Time calculation
    for (int i = 0; i < n; i++) {
        tat[i] = wt[i] + bt[i];
    }
    System.out.println("\nPID\tBT\tPriority\tWT\tTAT");
    for (int i = 0; i < n; i++) {
        System.out.println(pid[i] + "\t" + bt[i] + "\t" + pr[i] +
                "\t\t" + wt[i] + "\t" + tat[i]);
    }
    sc.close();
  }
}
```

OUPUT:

Enter number of processes: 3

Enter Burst Time for P1: 10

Enter Priority for P1: 3

Enter Burst Time for P2: 1

Enter Priority for P2: 1

Enter Burst Time for P3: 2

Enter Priority for P3: 2

```
                                        //5.BANKERS ALGORITHM
import java.util.*;
class Bankers
{
   int claim[][],alloc[][],av[],need[],p,r,alsum[],sum,temp[][];
   int a=0,b=0,c;
   boolean flag;
   Scanner src = new Scanner(System.in);
   Bankers(int p,int r)
   {
       this.p=p;
       this.r=r;
       claim=new int[p][r];
       alloc=new int[p][r];
       av=new int [r];
       need=new int [r];
       alsum=new int[r];
       temp=new int[p][r];
       System.out.println("ENTER THE "+r+" VALUES IN AVAILABLE VECTOR");
       for(int i=0;i<r;i++)
       {
          av[i]=src.nextInt();
       }
       System.out.println("ENTER THE CLAIM MATRIX OF"+p+" X "+r);
       for(int i=0;i<p;i++)
            for(int j=0;j<r;j++)
                 claim[i][j]=src.nextInt();
       System.out.println("ENTER THE ALLOCATION MATRIX OF "+p+" X "+r);
       for(int i=0;i<p;i++)
             for(int j=0;j<r;j++)
       alloc[i][j]=src.nextInt();
    }
    public void calNeed()
    {
       for(int i=0;i<r;i++)
       {
            for(int j=0;j<p;j++)
            {
               sum=sum+alloc[j][i];
            }
            alsum[i]=sum;
            sum=0;
       }
       for(int i=0;i<r;i++)
       {
          need[i]=av[i]-alsum[i];
          System.out.print("Initial Need Vector ");
          System.out.print(need[i]);
          System.out.println();
          System.out.println();
       }    }
```

```
    public void checkNeed()
    {
       for( int i=0;i<p;i++)
            for(int j=0;j<r;j++)
            {
                  temp[i][j]=claim[i][j]-alloc[i][j];
            }   }
    public void compareNeed()
    {
     while(b<p)
     {
      flag=false;
      for(int i=0;i<p;i++)
      {
        for(int j=0;j<r;j++)
        {
          if(temp[i][0]==89)
               break;
          if(temp[i][j]<=need[j])
            a++ ;
          else break;
        }
        if(a==r)
        {
          System.out.println("Process P"+(i+1)+" is Completed");
          addtoNeed(i);
          temp[i][0]=89;
          a=0;
          c++;
          flag=true;
        }
       if(flag==true)
          break;
      }
      b++;
     }
    }
    public void addtoNeed(int k)
    {
         int n=k;
         for(int j=0;j<r;j++)
         {
           need[j]=need[j]+alloc[n][j];
         }
          if(c==p)
               System.out.println("SYSTEM IS IN SAFE STATE");
         else
System.out.println("SYSTEM IS NOT IN SAFE STATE");
    }
 }
```

```
class BankerExp1
{
     public static void main(String args[])
     {
        Scanner src = new Scanner(System.in);
        System.out.println("ENTER THE NO. OF PROCESS");
        int p= src.nextInt();
        System.out.println("ENTER THE NO. OF RESOURCES");
        int r = src.nextInt();
        Bankers obj = new Bankers(p,r);
        obj.calNeed();
        obj.checkNeed();
        obj.compareNeed();
      }
}
```

```
OUTPUT
ENTER THE NO. OF PROCESS:4
ENTER THE NO. OF RESOURCES:3
ENTER THE 3 VALUES IN AVAILABLE VECTOR:9 3 6
ENTER THE CLAIM MATRIX OF 4 X 3
3 2 2
6 1 3
3 1 4
4 2 2
ENTER THE ALLOCATION MATRIX OF 4 X 3
1 0 0
5 1 1
2 1 1
0 0 2
Initial Need Vector 1
Initial Need Vector 1
Initial Need Vector 2
Process P2 is Completed
Process P1 is Completed
Process P3 is Completed
Process P4 is Completed
SYSTEM IS IN SAFE STATE
```

```
//6.DINING PHILOSOPHER PROBLEM
import java.io.*;
class Philo
{
int state[]=new int[5];
int thinking,eating,hungry;
void Op()
{
thinking=0;
eating=1;
hungry=2;
}
public void pickup(int i)
{
state[i]=2;
System.out.println("Philosopher "+i+" is hungry");
test(i);
if(state[i]==2)
System.out.println("Philosopher "+i+" is waiting");
}
public void putdown(int i)
{
 if(state[i]==1)
      {
      state[i]=0;
      System.out.println("Philosopher"+i+" is thinking");
test((i+1)%5);
test((i+5)%5);
      }
}

public void test(int i)
{
if(state[i]==2 && state[(i+1)%5]!=1 && state[(i+4)%5]!=1)
      {
      state[i]=1;
      System.out.println("philosopher "+i+" is eating");
      }
}
}
class Philosopher
{
      public static void main(String args[])throws IOException
{
Philo d=new Philo();
for(int i=0;i<5;i++)
d.pickup(i);
for(int i=0;i<5;i++)
```

```
d.putdown(i);
}
}

/* Output for dining philosophers problem

Philosopher 0 is hungry
philosopher 0 is eating
Philosopher 1 is hungry
Philosopher 1 is waiting
Philosopher 2 is hungry
philosopher 2 is eating
Philosopher 3 is hungry
Philosopher 3 is waiting
Philosopher 4 is hungry
Philosopher 4 is waiting
Philosopher0 is thinking
Philosopher2 is thinking
philosopher 3 is eating
Philosopher3 is thinking
philosopher 4 is eating
Philosopher4 is thinking*/
```

```
                        //7.producer and consumer problem
import java.util.Scanner;
public class Prod1{
public static void main(String args[]) {
        int[] buffer = new int[10];
        int bufsize, in, out, produce, consume, choice = 0;
        in = 0;
        out = 0;
        bufsize = 10;
       Scanner cin = new Scanner(System.in);
        while(choice != 3) {
        System.out.print("\n1. Produce  \n 2. Consume \n3. Exit");
        System.out.print("\nEnter your choice: ");
        choice = cin.nextInt();
        switch(choice) {
        case 1:
                if((in + 1) % bufsize == out) {
                System.out.print("\nBuffer is Full");
                }
                else {System.out.print("\nEnter the value: ");
                        produce = cin.nextInt();
                        buffer[in] = produce;
                        in = (in + 1) % bufsize;
                }break;
        case 2:  if(in == out) {
                System.out.print("\nBuffer is Empty");
                } else {
                consume = buffer[out];
                System.out.print("\nThe consumed value is " + consume);
                out = (out + 1) % bufsize;
                }break;
        }}
        }

}
```

```
/*OUTPUT
D:\Java>java Prod1
1. Produce
 2. Consume
3. Exit
Enter your choice: 1
Enter the value: 15
1. Produce
 2. Consume
3. Exit
Enter your choice: 1

Enter the value: 20
1. Produce
 2. Consume
3. Exit
Enter your choice: 1
Enter the value: 30
1. Produce
 2. Consume
3. Exit
Enter your choice: 2
The consumed value is 15
1. Produce
 2. Consume
3. Exit
Enter your choice: 2

The consumed value is 20
1. Produce
 2. Consume
3. Exit
Enter your choice:3
*/
```

```
                                        //8. Memory Allocation-Best Fit
import java.util.Scanner;
public class Best1 {
        public final static int MAX = 25;
        private static int[] bf = new int[MAX], ff = new int[MAX];
        public static void main(String args[]) {
                int[] frag = new int[MAX], b = new int[MAX], f = new int[MAX];
                int i, nb, nf, temp, lowest = 9999;
       Scanner cin = new Scanner(System.in);
        System.out.print("\nEnter the number of blocks:");
        nb = cin.nextInt();
        System.out.print("Enter the number of files:");
        nf = cin.nextInt();
        System.out.println("\nEnter the size of the blocks:-");
        for(i = 1; i <= nb; i++) {
        System.out.println("Block " + i + ":");
                b[i] = cin.nextInt();
}               System.out.println("Enter the size of the files :-");
                for(i = 1; i <= nf; i++) {
                        System.out.print("File " + i + ":");
                        f[i] = cin.nextInt();
                }
for(i = 1; i <= nf; i++) {
for(int j = 1; j <= nb; j++) {
if(bf[j] != 1) {
                temp = b[j] - f[i];
                if(temp >= 0) {
                if(lowest > temp) {
                ff[i] = j;
                lowest = temp;
                        }
        }
}
}
frag[i] = lowest;
bf[ff[i]] = 1;
lowest = 9999;
}
System.out.print("\nFile No\tFile Size \tBlock No\tBlock Size\tFragment");
for(i = 1; i <= nf && ff[i] != 0; i++) {
System.out.print("\n" + i + "\t\t" + f[i] + "\t\t" + ff[i] + "\t\t" + b[ff[i]] + "\t\t" + frag[i]);
    }
  }
}
```

```
/*OUTPUT
D:\Java>javac Best1.java

D:\Java>java Best1

Enter the number of blocks:3
Enter the number of files:3

Enter the size of the blocks:-
Block 1:
100
Block 2:
200
Block 3:
50
Enter the size of the files :-
File 1:40
File 2:150
File 3:100

File No File Size      Block No      Block Size     Fragment
1           40          3           50           10
2           150          2           200          50
3           100          1           100          0
```

```
                        //9.Memory Fragmentation
import java.util.Scanner;
public class Mft {
        public static void main(String[] args) {
        int ms, bs, nob, ef, n, tif = 0;
        int[] mp = new int[10];
        int i, p = 0;
    Scanner cin = new Scanner(System.in);
        System.out.print("Enter the total memory available (in Bytes) -- ");
        ms = cin.nextInt();
        System.out.print("Enter the block size (in Bytes) -- ");
        bs = cin.nextInt();
        nob = ms / bs;
        ef = ms - nob * bs;
        System.out.print("\nEnter the number of processes -- ");
        n = cin.nextInt();
        for(i = 0; i < n; i++) {
        System.out.print("Enter memory required for process " + (i + 1) + " (in Bytes)-- ");
        mp[i] =cin.nextInt();
        }
        System.out.print("\nNo. of Blocks available in memory -- " + nob);
        System.out.print("\n\nPROCESS\tMEMORY REQUIRED\t ALLOCATED\tINTERNAL
FRAGMENTATION");
        for(i = 0; i < n && p < nob; i++) {
        System.out.print("\n " + (i + 1) + "\t\t" + mp[i]);
        if(mp[i] > bs) {            System.out.print("\t\tNO\t\t---");
                } else {System.out.print("\t\tYES\t" + (bs - mp[i]));
        tif = tif + bs - mp[i];
                p++;
                        }
                }
if(i < n)
 {System.out.print("\nMemory is Full, Remaining Processes cannot be accomodated");
                }
System.out.print("\n\nTotal Internal Fragmentation is " + tif);
                System.out.print("\nTotal External Fragmentation is " + ef);
        }
}
```

```
/*OUTPUT
D:\Java>java Mft
Enter the total memory available (in Bytes) -- 1000
Enter the block size (in Bytes) -- 300
Enter the number of processes -- 5
Enter memory required for process 1 (in Bytes)-- 260
Enter memory required for process 2 (in Bytes)-- 400
Enter memory required for process 3 (in Bytes)-- 300
Enter memory required for process 4 (in Bytes)-- 150
Enter memory required for process 5 (in Bytes)-- 230
No. of Blocks available in memory -- 3

PROCESS MEMORY REQUIRED  ALLOCATED     INTERNAL FRAGMENTATION
 1          260          YES    40
 2          400          NO           ---
 3          300          YES    0
 4          150          YES    150
Memory is Full, Remaining Processes cannot be accomodated

Total Internal Fragmentation is 190
Total External Fragmentation is 100*/
```

//10.FIFO PAGE REPLACEMENT

```
import java.io.*;
public class FIFO12 {

   public static void main(String[] args) throws IOException
   {
      BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
      int frames, pointer = 0, hit = 0, fault = 0,ref_len;
      int buffer[];
      int reference[];
      int mem_layout[][];

      System.out.println("Please enter the number of Frames: ");
      frames = Integer.parseInt(br.readLine());

      System.out.println("Please enter the length of the Reference string: ");
      ref_len = Integer.parseInt(br.readLine());

      reference = new int[ref_len];
      mem_layout = new int[ref_len][frames];
      buffer = new int[frames];
      for(int j = 0; j < frames; j++)
            buffer[j] = -1;

      System.out.println("Please enter the reference string: ");
      for(int i = 0; i < ref_len; i++)
      {
         reference[i] = Integer.parseInt(br.readLine());
      }
      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++;
         break;
        }       }
       if(search == -1)
       {
        buffer[pointer] = reference[i];
        fault++;
        pointer++;
        if(pointer == frames)
         pointer = 0;
```

```
        }
          for(int j = 0; j < frames; j++)
             mem_layout[i][j] = buffer[j];
        }

        for(int i = 0; i < frames; i++)
        {
          for(int j = 0; j < ref_len; j++)
             System.out.printf("%3d ",mem_layout[j][i]);
          System.out.println();
        }

        System.out.println("The number of Hits: " + hit);
        System.out.println("Hit Ratio: " + (float)((float)hit/ref_len));
        System.out.println("The number of Faults: " + fault);
    }

}
/*OUTPUT:-
Please enter the number of Frames: 3
Please enter the length of the Reference string: 20
Please enter the reference string:
7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

  7  7  7  2  2  2  2  4  4  4  0  0  0  0  0  0  0  7  7  7
 -1  0  0  0  0  3  3  3  2  2  2  2  2  1  1  1  1  1  0  0
 -1 -1  1  1  1  1  0  0  0  3  3  3  3  3  2  2  2  2  2  1
The number of Hits: 5
Hit Ratio: 0.25
The number of Faults: 15--------------------------------*/
```

//11.LRU PAGE REPLACEMENT

```
import java.util.*;
class LRU
{
	Scanner sc=new Scanner(System.in);
	int[] frame,page,present;
	int size,pages,pf=0,flag=0,flag1=0;
	LRU(int size,int pages)
	{
		this.size=size;
		this.pages=pages;
		frame=new int[size];
		present=new int[size];
		page=new int[pages];}
	void get()
	{
		System.out.println("Enter pages");
		for(int i=0;i<pages;i++)
			page[i]=sc.nextInt();
		for(int i=0;i<size;i++)
			frame[i]=-1;}
	int check(int x)
	{
		flag=-1;
		for(int i=0;i<size;i++)
			if(frame[i]==x)
			{
				flag=i;
				break;}
		return flag;}
	int replace(int x)
	{
		int i;
		for(i=0;i<size;i++)
			present[i]=0;
		flag1=0;
		for(i=x-1;i>=0;i--)
		{
			if(check(page[i])>=0)
			{
				flag1++;
				present[check(page[i])]=1;}
			if(flag1==(size-1)) break;
		}
		for(i=0;i<size;i++)
			if(present[i]==0)
			{
```

```
                        flag1=i;
                        break;}
                return i;
        }
        void lru()
        {
                for(int i=0;i<pages;i++)
                {
                        if(i<size)
                        {
                                frame[i]=page[i];
                                pf++;
                                for(int j=0;j<size;j++)
                                        System.out.print(frame[j]+"    ");
                                System.out.println();}
                        else
                        {
                                if(check(page[i])==-1)
                                {
                                        int r=replace(i);
                                        frame[r]=page[i];
                                        pf++;
                                        for(int j=0;j<size;j++)
                                                System.out.print(frame[j]+"    ");
                                        System.out.println();}
                                else
                                {
                                        for(int j=0;j<size;j++)
                                                System.out.print(frame[j]+"    ");
                                        System.out.println();}}}
                System.out.println("PAGE FAULT: "+pf);}}
class KLRU
{
        public static void main(String arg[])
        {
                Scanner s=new Scanner(System.in);
                System.out.print("Enter frame size:");
                int n=s.nextInt();
                System.out.print("Enter no of pages:");
                int p=s.nextInt();
                LRU obj=new LRU(n,p);
                obj.get();
                obj.lru();
}
}
```

OUTPUT

```
D:\Java>java KLRU
Enter frame size:3
Enter no of pages:20
Enter pages
7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1
7   -1   -1
7   0   -1
7   0   1
2   0   1
2   0   1
2   0   3
2   0   3
4   0   3
4   0   2
4   3   2
0   3   2
0   3   2
0   3   2
1   3   2
1   3   2
1   0   2
1   0   2
1   0   7
1   0   7
1   0   7
PAGE FAULT: 12

D:\Java>
```

//12. OPTIMAL PAGE REPLACEMENT

```
import java.io.*;
import java.util.*;
public class Opt12{
   public static void main(String[] args) throws IOException
   {
      Scanner br = new Scanner(System.in);
      int frames, pointer = 0, hit = 0, fault = 0,ref_len;
      boolean isFull = false;
      int buffer[];
      int reference[];
      int mem_layout[][];

      System.out.println("Please enter the number of Frames: ");
      frames = br.nextInt();

      System.out.println("Please enter the length of the Reference string: ");
      ref_len = br.nextInt();

      reference = new int[ref_len];
      mem_layout = new int[ref_len][frames];
      buffer = new int[frames];
      for(int j = 0; j < frames; j++)
            buffer[j] = -1;

      System.out.println("Please enter the reference string: ");
      for(int i = 0; i < ref_len; i++)
      {
         reference[i] = br.nextInt();
      }
      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++;
         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];
 fault++;
 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 < frames; i++)
{
   for(int j = 0; j < ref_len; j++)
```

```
            System.out.printf("%3d ",mem_layout[j][i]);
          System.out.println();
       }

       System.out.println("The number of Hits: " + hit);
       System.out.println("Hit Ratio: " + (float)((float)hit/ref_len));
       System.out.println("The number of Faults: " + fault);
    }
}
/*OUTPUT:

D:\Java>java Opt12
Please enter the number of Frames:
3
Please enter the length of the Reference string:
20
Please enter the reference string:
7 0 1 2 0 3 0 4 2 3 0 3 2 1 2 0 1 7 0 1

 7  7  7  2  2  2  2  2  2  2  2  2  2  2  2  2  7  7  7

-1  0  0  0  0  0  0  4  4  4  0  0  0  0  0  0  0  0  0  0

-1 -1  1  1  1  3  3  3  3  3  3  3  3  1  1  1  1  1  1  1

The number of Hits: 11
Hit Ratio: 0.55
The number of Faults: 9 */
```

```
                                        //13.SCAN Disk Scheduling Algorithm
import java.util.*;
public class Scan1
{       public static void main(String args[])throws IOException
  {
        Scanner br=new Scanner(System.in);
        int a,x,c,h,i,hm=0,temp;
        int ar[]=new int[30];
        System.out.println("\nSCAN Scheduling");
        System.out.println("\nEnter no of process:");
        a=br.nextInt();
        System.out.println("\ninitial head movemnt:");
        h=br.nextInt();
        System.out.println("\n");
        ar[0]=0;
        for(c=1;c<=a;c++)
        {  System.out.println("request"+c+":");
        ar[c]=br.nextInt();
        }
        ar[c]=h;
        for(c=0;c<a+1;c++)
        {   for(i=0;i<a-c+1;i++)
                {if(ar[i]>ar[i+1])
                  {   temp=ar[i];
                        ar[i]=ar[i+1];
                        ar[i+1]=temp;
                   }
                }}
        for(c=0;c<a+1;c++)
         {   x=ar[c+1]-ar[c];
                hm=hm+x;
                if(ar[c]==h)
                { hm=hm-x;
                  hm=hm+ar[c+1];
                }
                else;
        }
        System.out.println("\ntotal movements:"+hm);
   }}


/*OUTPUT:
D:\Java>java Scan12
Enter no of process: 8
initial head movement: 53
request1:98
request2:183
request3:37
```

request4:122
request5:14
request6:124
request7:65
request8:67
total movements:236
***/**

//14.CLOOK SCHEDULING

```
import java.io.*;
import java.util.*;
public class Clook
{
 public static void main(String args[])throws IOException
{ Scanner s=new Scanner(System.in);
        int a,b,c,k=0,d=1,i,hm=0,head,temp;
        int ar[]=new int[30];
        System.out.println("\n Clook disk Scheduling");
        System.out.println("\n How many process:");
        a=s.nextInt();
        System.out.println("\n initial head position:");
        b=s.nextInt();
        System.out.println(" enter request");
        for(c=0;c<a;c++)
        {System.out.println(" request:"+d+":");
        ar[c]=s.nextInt();
        d++;
        }
        System.out.println(" Computation");
        ar[c]=b;
        for(c=0;c<=a-1;c++)
        { for(i=0;i<=a-1-c;i++)
                {if(ar[i]>ar[i+1])
                 { temp=ar[i];
                   ar[i]=ar[i+1];
                  ar[i+1]=temp;
                 }
                }
        }
        for(c=0;c<a;c++)
        {       if(ar[c]==b)
                { k=c;
                }
        }
        for(c=k;c<a;c++)
        {       head=ar[c]-ar[c+1];
                if(head<0)
                {       head=head*(-1);
                }
        System.out.print("|"+ar[c]+"-"+ar[c+1]+"|"+"+");
        hm=hm+head;
        }
        head=ar[c]-ar[0];
        hm=hm+head;
        System.out.print("|"+ar[c]+"-"+ar[0]+"|");
        for(c=0;c<k-1;c++)
```

```
        { head=ar[c]-ar[c+1];
        if(head<0)
        {       head=head*(-1);
        }
        if(c<k-1)
        {System.out.println(" +");
        }
        hm=hm+head;
        System.out.println("|"+ar[c]+"-"+ar[c+1]+"|");
        }
System.out.println("="+hm);
System.out.println("\n total head moves:"+hm);
}
}
/*OUTPUT:

D:\Java>java Clook

 Clook disk Scheduling

 How many process:8

 initial head position:53
 enter request
 request:1:98
 request:2:183
 request:3:37
 request:4:122
 request:5:14
 request:6:124
 request:7:65
 request:8:67
 Computation
|53-65|+|65-67|+|67-98|+|98-122|+|122-124|+|124-183|+|183-14| +
|14-37|
=322

 total head moves:322  */
```

```
//15.FCFS Scheduling
import java.util.*;
class DiskF
{
	public static void main(String args[])
{
	int n,i,hs,flag,total=0;
	Scanner sc=new Scanner(System.in);
	System.out.println("Enter no of cylinders");
	n=sc.nextInt();
	int c[]=new int[n];
	int hm[]=new int[n];
	System.out.println("Enter the cylinder no.");
	for(i=0;i<n;i++)
	c[i]=sc.nextInt();
	System.out.println("enter head start");
	hs=sc.nextInt();
	flag=hs;
	for(i=0;i<n;i++)
	{
	if(c[i] > flag)
	hm[i]=c[i]-flag;
	else
	hm[i]=flag-c[i];
	flag=c[i];
	}
	for(i=0;i<n;i++)
	total=total+hm[i];
	System.out.println("The total no of cylinders traversed during disk movement is:"+ total);
	}
}
/*OUTPUT:

D:\Java>java DiskF
Enter no of cylinders
8
Enter the cylinder no.
98  183  37  122  14  124  65 67
enter head start
53
The total no of cylinders traversed during disk movement is:640
*/
```