

1)

### Code

```
import java.util.*;
import java.io.*;
public class Pass1
{
    static int address=0;
    static int sadd[]=new int[10];
    static int ladd[]=new int[10];
    public static void main(String args[])
    {

        BufferedReader br;
        OutputStream oo;
        String input=null;

        String IS[]={"ADD","SUB","MUL","MOV"};
        String UserReg[]={"AREG","BREG","CREG","DREG"};
        String AD[]={"START","END"};
        String DL[]={"DC","DS"};
        int lc=0;
        int scount=0,lcount=0;
        int flag=0,flag2=0,stored=0;

        String tokens[]=new String[30];
        String tt=null;

        String sv[]=new String[10];
        String lv[]=new String[10];

        try
        {
            br=new BufferedReader(new
FileReader("initial.txt"));
            File f = new File("IM.txt");
            File f1 = new File("ST.txt");
            File f2 = new File("LT.txt");
            PrintWriter p = new PrintWriter(f);
            PrintWriter p1 = new PrintWriter(f1);
            PrintWriter p2 = new PrintWriter(f2);
            int k=0,l=0;
            while ((input = br.readLine()) != null)
            {
                StringTokenizer st = new
StringTokenizer(input," ");
                while (st.hasMoreTokens())
                {

                    tt=st.nextToken();

                    //System.out.println(tt);
```

```

if(tt.matches("\\d*") && tt.length() > 2)
{

lc=Integer.parseInt(tt);
p.println(lc);
address=lc-1;

}
else
{

    for(int i=0;i<AD.length;i++)
    {
        if(tt.equals(AD[i]))
        {
            p.print("AD "+(i+1)+" ");
        }

    }

    for(int i=0;i<IS.length;i++)
    {
        if(tt.equals(IS[i]))
        {
            p.print("IS "+(i+1)+" ");
        }

    }

    for(int i=0;i<UserReg.length;i++)
    {
        if(tt.equals(UserReg[i]))
        {
            p.print((i+1)+" ");
            flag=1;

```

```

}

}

for(int i=0;i<DL.length;i++)

{

    if(tt.equals(DL[i]))

    {

        p.print("DL "+(i+1)+" ");

    }

}

if(tt.length()==1 && !(st.hasMoreTokens()) && flag==1)

{

    if ( Arrays.asList(sv).contains(tt) )

    {

        for(int i=0;i<scount;i++)

        {

            if(sv[i].equals(tt))

            {

                p.print("S"+i);

                flag2=1;

            }

            else

            {

                flag2=0;

            }

        }

    }

}

```

```

}
else
{
    p.print("S"+scount);

    sv[scount]=tt;

    flag2=1;

    scount++;

}

}

if(tt.length()==1 && (st.hasMoreTokens()))
{
    p.print(tt+" ");

    sadd[k]=address;k++;

}

if(tt.charAt(0)=='=')
{
    p.print("L"+lcount);

    lv[lcount]=tt;

    lcount++;

}

if(!st.hasMoreTokens())
{
    p.println();

}

```

```

if(tt.equals("DS"))
{
    int a=Integer.parseInt(st.nextToken());

    address=address+a-1;

    p.println();
}

}

}

        }
        //System.out.println();
        address++;
    } p.close();
    address--;

    for(int i=0;i<lcount;i++)
    {
        ladd[i]=address;
        address++;
    }

    for(int i=0;i<scount;i++)
    {
        p1.println(i+"\t"+sv[i]+" \t"+sadd[i]);
    }p1.close();

    for(int i=0;i<lcount;i++)
    {
        p2.println(i+"\t"+lv[i]+" \t"+ladd[i]);
    }p2.close();
}
catch(Exception e)
{
    e.printStackTrace();
}    }    }

```

```
Welcome | J Pass1.java 5 | IM.txt | initial.txt | LT.txt | J Pass1.class | ...
LP_code > IM.txt
1 AD 1 100
2 IS 4 1 S0
3 IS 4 2 S1
4 IS 4 3 L0
5 IS 4 4 L1
6 IS 1 1 2
7 IS 2 1 S0
8 A DL 1
9 B DL 2
10 AD 2
11

PROBLEMS (5) | OUTPUT | DEBUG CONSOLE | TERMINAL | PORTS
Error: Could not find or load main class Pass1.java
PS D:\engineering\TE\spos\LP_code> java Pass1
Error: Could not find or load main class Pass1
PS D:\engineering\TE\spos\LP_code> javac Pass1.java
PS D:\engineering\TE\spos\LP_code> java Pass1
PS D:\engineering\TE\spos\LP_code> 
```

```
LP_code > ST.txt
1 0 A 106
2 1 B 107
3
```

```
LP_code > LT.txt
1 0 =2 110
2 1 =3 111
3
```

LP\_code > ≡ initial.txt

```
1  START 100
2  MOV AREG A
3  MOV BREG B
4  MOV CREG =2
5  MOV DREG =3
6  ADD AREG BREG
7  SUB AREG A
8  A DC 05
9  B DS 03
10 END
```

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

Error: Could not find or load main class Pass1.java

PS D:\engineering\TE\spos\LP\_code> java Pass1

Error: Could not find or load main class Pass1

PS D:\engineering\TE\spos\LP\_code> javac Pass1.java

PS D:\engineering\TE\spos\LP\_code> java Pass1

PS D:\engineering\TE\spos\LP\_code>

## 2)Macro Code

```
import java.util.*;
import java.io.*;
public class Macro
{
    public static void main(String args[])
    {

        BufferedReader br;
        OutputStream oo;
        String input=null;
        String tt=null;
        String arg=null;
        String macroTokens=null;
        String mnt[]=new String[10];
        String mdt[]=new String[20];
        String AR[]=new String[20];
        int macroindex[]=new int[10];
        int mcount=0,arg_count=0;
        int middlecount=0;
        int index=1;
        int macro_enc=0;

        try
        {
            br=new BufferedReader(new
FileReader("Input.txt"));
            File f3 = new File("mnt.txt");
            File f4 = new File("mdt.txt");
            File f5 = new File("adt.txt");
            PrintWriter p3 = new PrintWriter(f3);
            PrintWriter p4 = new PrintWriter(f4);
            PrintWriter p5 = new PrintWriter(f5);
            while ((input = br.readLine()) != null)
            {
                StringTokenizer st = new StringTokenizer(input," ");
                tt=st.nextToken();
                if(tt.equals("MACRO"))
                {
                    macro_enc=1;
                    tt=st.nextToken();
                    mnt[mcount]=tt;
                }
            }
        }
    }
}
```



```

        macroindex[mcount]=index;

p3.println(mnt[mcount]+"\\t"+macroindex[mcount]);
p4.println(mnt[mcount]);
p5.println(mnt[mcount]);
mcount++;

tt=st.nextToken();
StringTokenizer t = new
StringTokenizer(tt, ",");
while (t.hasMoreTokens())
{
    arg=t.nextToken();
    if(arg.charAt(0)=='&')
    {
        AR[arg_count]=arg;
        p5.println(AR[arg_count]);
        arg_count++;
    }
}

else
{
    if(macro_enc==1)
    {
        if(input.equals("MEND"))
        {
            macro_enc=0;
            p4.println("MEND");
        }
        else
        {
            StringTokenizer t=new
StringTokenizer(input, " ");

            while(t.hasMoreTokens())
            {
                macroTokens=t.nextToken();
                for(int

i=0;i<arg_count;i++)

                {

                    if(macroTokens.charAt(0)=='&' && macroTokens.equals(AR[i]))

```

```

        {

p4.print("AR"+i);

        }
    }

    if (macroTokens.charAt(0)=='&') {}

    else

        {

p4.print(macroTokens+" ");

        }

    if (!t.hasMoreTokens())

        {

p4.println();

        }

    }

}

}

index++;
}
p3.close();
p4.close();
p5.close();

}

catch (Exception e)
{

    e.printStackTrace();

}

}

}

```

≡ adt.txt

1 INCR1

2 &FIRST

3 &SECOND

4 INCR2

5 &ARG1

6 &ARG2

7

▼ SPOS

≡ adt.txt

≡ Input.txt

J Macro.class

J Macro.java

≡ mdt.txt

≡ mnt.txt

> OUTLINE

> TIMELINE

> JAVA PROJECTS

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\engineering\TE\spos> javac Macro.java  
PS D:\engineering\TE\spos> java Macro  
PS D:\engineering\TE\spos>

≡ Input.txt

1 MACRO INCR1 &FIRST,&SECOND

2 ADD AREG &FIRST

3 LDA BREG &SECOND

4 MEND

5 MACRO INCR2 &ARG1,&ARG2

6 MOV CREG &ARG1

7 SUB DREG &ARG2

8 MEND

9 START 100

10 MOV AREG A

11 MOV BREG B

12 INCR1

13 MOV CREG =2

14 MOV DREG =3

15 ADD AREG BREG

16 A DC 05

17 B DS 03

18 END

▼ SPOS

≡ adt.txt

≡ Input.txt

J Macro.class

J Macro.java

≡ mdt.txt

≡ mnt.txt

> OUTLINE

> TIMELINE

> JAVA PROJECTS

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\engineering\TE\spos> javac Macro.java  
PS D:\engineering\TE\spos> java Macro  
PS D:\engineering\TE\spos>

mdt.txt

1 INCR1  
2 ADD AREG AR0  
3 LDA BREG AR1  
4 MEND  
5 INCR2  
6 MOV CREG AR2  
7 SUB DREG AR3  
8 MEND  
9

SPOS

adt.txt  
Input.txt  
Macro.class  
Macro.java  
mdt.txt  
mnt.txt

OUTLINE  
TIMELINE  
JAVA PROJECTS

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\engineering\TE\spos> javac Macro.java  
PS D:\engineering\TE\spos> java Macro  
PS D:\engineering\TE\spos>

Input.txt mdt.txt mnt.txt

mnt.txt

1 INCR1 1  
2 INCR2 5  
3

SPOS

adt.txt  
Input.txt  
Macro.class  
Macro.java  
mdt.txt

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS D:\engineering\TE\spos> javac Macro.java  
PS D:\engineering\TE\spos> java Macro  
PS D:\engineering\TE\spos>

5)Write a program to simulate CPU Scheduling Algorithms: FCFS, SJF (Preemptive), Priority (Non- Preemptive) and Round Robin (Preemptive).

```
import java.util.*;
import java.io.*;

public class Fcfs
{

    public static void main(String args[])
    {

        public static void main(String args[])
        {
            int n,sum=0;
            float total_tt=0,total_waiting=0;

            Scanner s=new Scanner(System.in);
            System.out.println("Enter Number Of Process you want to
Execute---");
            n=s.nextInt();
            int arrival[]=new int[n];
            int cpu[]=new int[n];
            int finish[]=new int[n];
            int turntt[]=new int[n];
            int wait[]=new int[n];
            int process[]=new int[n];

            // int pro[][]=new int[3][3];
            for(int i=0;i<n;i++)
            {
                System.out.println("Enter arrival time of "+(i+1)+"
Process : ");
                arrival[i]=s.nextInt();
                System.out.println("Enter CPU time of "+(i+1)+" Process
: ");
                cpu[i]=s.nextInt();

                process[i]=i+1;
            }

            for(int i=0;i<n;i++)
            {
                sum=sum+cpu[i];
                finish[i]=sum;
            }
        }
    }
}
```

```

        for(int i=0;i<n;i++)
        {
            turntt[i]=finish[i]-arrival[i];

            total_tt=total_tt+turntt[i];

            wait[i]=turntt[i]-cpu[i];

            total_waiting+=wait[i];
        }

        System.out.println("\n\nProcess\t\tAT\tCPU_T");
        for(int i=0;i<n;i++)
        {

System.out.println(process[i]+\t\t+arrival[i]+\t\t+cpu[i]);
        }

        System.out.println("\n\n");
        System.out.println("Total turn around time is :
"+(total_tt/n));
        System.out.println("Total waiting time is :
"+(total_waiting/n));

    }
}

```

```

J Fcfs.java > Fcfs > main(String[])
    int n,sum=0;
10     float total_tt=0,total_waiting=0;
11
12     Scanner s=new Scanner(System.in);
13     System.out.println(x:"Enter Number Of Process you want to Execute---");

```

PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

PS D:\engineering\TE\spos> javac Fcfs.java
PS D:\engineering\TE\spos> java Fcfs
Enter Number Of Process you want to Execute---
4
Enter arrival time of 1 Process :
0
Enter CPU time of 1 Process :
8
Enter arrival time of 2 Process :
1
Enter CPU time of 2 Process :
4
Enter arrival time of 3 Process :
2
Enter CPU time of 3 Process :
9
Enter arrival time of 4 Process :
3
Enter CPU time of 4 Process :
5

```

Process	AT	CPU_T
1	0	8
2	1	4
3	2	9
4	3	5

```

Total turn around time is : 15.25
Total waiting time is : 8.75
PS D:\engineering\TE\spos>

```

```

import java.util.*;
import java.io.*;

public class Robbin
{

    public static void main(String args[])
    {

        int n,sum=0;
        float total_tt=0,total_waiting=0;

        Scanner s=new Scanner(System.in);
        System.out.println("Enter Number Of Process you want to
Execute---");
        n=s.nextInt();
        int arrival[]=new int[n];
        int cpu[]=new int[n];
        int ncpu[]=new int[n];
        int pri[]=new int[n];
        int finish[]=new int[100];
        int turntt[]=new int[n];
        int wait[]=new int[n];
        int process[]=new int[n];
        int t_quantum,difference,temp_sum=0,k=0;
        int seq[]=new int[100];

        // int pro[][]=new int[3][3];
        for(int i=0;i<n;i++)
        {

            System.out.println("Enter arrival time of "+(i+1)+"
Process : ");
            arrival[i]=s.nextInt();
            System.out.println("Enter CPU time of "+(i+1)+" Process
: ");
            ncpu[i]=cpu[i]=s.nextInt();

            process[i]=i+1;
        }

        System.out.println("Enter time quantum : ");
        t_quantum = s.nextInt();
    }
}

```



```

int tv=0;
for(int i=0;i<n;i++){temp_sum=temp_sum+cpu[i];}
//System.out.println(temp_sum);

System.out.println("Process execution sequence : ");
while(sum!=temp_sum){
for(int i=0;i<n;i++)
{
    if(ncpu[i]<t_quantum)
    {
        difference=ncpu[i];
        tv=ncpu[i];
        ncpu[i]=0;
    }
    else
    {
        difference = ncpu[i]-t_quantum;
        tv=t_quantum;
        ncpu[i]=difference;
    }
    if(tv > 0)
    {
        sum=sum+tv;
        finish[k]=sum;
        seq[k]=i;
        System.out.print(seq[k]+1+" ");

        k++;

    }
}
}
System.out.println();

for(int i=0;i<n;i++)
{
    int carr=0,tt=0;
    carr=arrival[i];

    for(int j=0;j<k;j++)
    {

```

```

        if(seq[j]==i)
        {
            tt=tt+(finish[j]-carr);
            carr=finish[j];
        }
    }

    turntt[i]=tt;
    System.out.println("Turn around time for "+(i+1)+"
process : "+turntt[i]);
    total_tt=total_tt+turntt[i];

    wait[i]=turntt[i]-cpu[i];

    System.out.println("Waiting time for "+(i+1)+" process
: "+wait[i]);

    total_waiting+=wait[i];
}

System.out.println("\n\nProcess\t\tAT\tCPU_T");
for(int i=0;i<n;i++)
{

System.out.println(process[i]+\t\t+arrival[i]+\t\t+cpu[i]);
}

    System.out.println("\n\n");
    System.out.println("Total turn around time is :
"+(total_tt/n));
    System.out.println("Total waiting time is :
"+(total_waiting/n));

}
}

```

```
PS D:\engineering\TE\spos> javac Robbin.java
PS D:\engineering\TE\spos> java Robbin
Enter Number Of Process you want to Execute---
```

```
4
Enter arrival time of 1 Process :
```

```
0
Enter CPU time of 1 Process :
```

```
8
Enter arrival time of 2 Process :
```

```
1
Enter CPU time of 2 Process :
```

```
4
Enter arrival time of 3 Process :
```

```
2
Enter CPU time of 3 Process :
```

```
9
Enter arrival time of 4 Process :
```

```
3
Enter CPU time of 4 Process :
```

```
5
Enter time quantum :
```

```
2
Process execution sequence :
1 2 3 4 1 2 3 4 1 3 4 1 3 3
Turn around time for 1 process : 23
Waiting time for 1 process : 15
Turn around time for 2 process : 11
Waiting time for 2 process : 7
Turn around time for 3 process : 24
Waiting time for 3 process : 15
Turn around time for 4 process : 18
Waiting time for 4 process : 13
```

Process	AT	CPU_T
1	0	8
2	1	4
3	2	9
4	3	5

```
Total turn around time is : 19.0
Total waiting time is : 12.5
PS D:\engineering\TE\spos>
```

```

import java.util.*;
import java.io.*;

public class Priority{

    public static void main(String args[])
    {
        int n,sum=0;
        float total_tt=0,total_waiting=0;

        Scanner s=new Scanner(System.in);
        System.out.println("Enter Number Of Process U want 2
Execute---");
        n=s.nextInt();
        int arrival[]=new int[n];
        int cpu[]=new int[n];
        int pri[]=new int[n];
        int finish[]=new int[n];
        int turntt[]=new int[n];
        int wait[]=new int[n];
        int process[]=new int[n];

        // int pro[][]=new int[3][3];
        for(int i=0;i<n;i++)
        {
            System.out.println("Enter arrival time of "+(i+1)+"
Process : ");
            arrival[i]=s.nextInt();
            System.out.println("Enter CPU time of "+(i+1)+" Process
: ");
            cpu[i]=s.nextInt();

            System.out.println("Enter Priority of "+(i+1)+" Process
: ");
            pri[i]=s.nextInt();

            process[i]=i+1;
        }

        for(int i=0;i<n-1;i++)
        {
            for(int j=i+1;j<n;j++)

```

```

        {
            if(pri[i]>pri[j])
            {
                int temp=cpu[i];
                cpu[i]=cpu[j];
                cpu[j]=temp;

                //temp=arrival[i];
                //arrival[i]=arrival[j];
                //arrival[j]=temp;

                temp=process[i];
                process[i]=process[j];
                process[j]=temp;

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

    for(int i=0;i<n;i++)
    {
        sum=sum+cpu[i];
        finish[i]=sum;
    }

    for(int i=0;i<n;i++)
    {
        turntt[i]=finish[i]-arrival[i];

        total_tt=total_tt+turntt[i];

        wait[i]=turntt[i]-cpu[i];

        total_waiting+=wait[i];
    }

    System.out.println("\n\nProcess\t\tAT\tCPU_T");
    for(int i=0;i<n;i++)

```

```
        {  
  
        System.out.println(process[i]+"\\t\\t"+arrival[i]+"\\t"+cpu[i]);  
        }  
  
        System.out.println("\\n\\n");  
        System.out.println("Total turn around time is :  
"+(total_tt/n));  
        System.out.println("Total waiting time is :  
"+(total_waiting/n));  
  
    }  
}
```

```
PS D:\engineering\TE\spos> javac Priority.java
```

```
PS D:\engineering\TE\spos> java Priority
```

```
Enter Number Of Process U want 2 Execute---
```

```
4
```

```
Enter arrival time of 1 Process :
```

```
0
```

```
Enter CPU time of 1 Process :
```

```
8
```

```
Enter Priority of 1 Process :
```

```
3
```

```
Enter arrival time of 2 Process :
```

```
1
```

```
Enter CPU time of 2 Process :
```

```
4
```

```
Enter Priority of 2 Process :
```

```
2
```

```
Enter arrival time of 3 Process :
```

```
2
```

```
Enter CPU time of 3 Process :
```

```
9
```

```
Enter Priority of 3 Process :
```

```
1
```

```
Enter arrival time of 4 Process :
```

```
3
```

```
Enter CPU time of 4 Process :
```

```
5
```

```
Enter Priority of 4 Process :
```

```
4
```

Process	AT	CPU_T
3	0	9
2	1	4
1	2	8
4	3	5

```
Total turn around time is : 15.75
```

```
Total waiting time is : 9.25
```

```

import java.util.*;
import java.io.*;

public class Sfj
{

    public static void main(String args[])
    {

        int n,sum=0;
        float total_tt=0,total_waiting=0;

        Scanner s=new Scanner(System.in);
        System.out.println("Enter Number Of Process U want 2
Execute---");
        n=s.nextInt();
        int arrival[]=new int[n];
        int cpu[]=new int[n];
        int finish[]=new int[n];
        int turntt[]=new int[n];
        int wait[]=new int[n];
        int process[]=new int[n];

        // int pro[][]=new int[3][3];
        for(int i=0;i<n;i++)
        {
            System.out.println("Enter arrival time of "+(i+1)+"
Process : ");
            arrival[i]=s.nextInt();
            System.out.println("Enter CPU time of "+(i+1)+" Process
: ");
            cpu[i]=s.nextInt();

            process[i]=i+1;
        }

        for(int i=0;i<n-1;i++)
        {
            for(int j=i+1;j<n;j++)
            {
                if(cpu[i]>cpu[j])
                {
                    int temp=cpu[i];

```



```

        cpu[i]=cpu[j];
        cpu[j]=temp;

        temp=arrival[i];
        arrival[i]=arrival[j];
        arrival[j]=temp;

        temp=process[i];
        process[i]=process[j];
        process[j]=temp;

    }

}

for(int i=0;i<n;i++)
{
    sum=sum+cpu[i];
    finish[i]=sum;
}

for(int i=0;i<n;i++)
{
    turntt[i]=finish[i]-arrival[i];

    total_tt=total_tt+turntt[i];

    wait[i]=turntt[i]-cpu[i];

    total_waiting+=wait[i];
}

System.out.println("\n\nProcess\t\tAT\tCPU_T");
for(int i=0;i<n;i++)
{

System.out.println(process[i]+\t\t"+arrival[i]+\t"+cpu[i]);

}

System.out.println("\n\n");
System.out.println("Total turn around time is :
" +(total_tt/n));

```

```
        System.out.println("Total waiting time is :  
"+(total_waiting/n));  
  
    }  
}
```

```
PS D:\engineering\TE\spos> javac Sfj.java  
PS D:\engineering\TE\spos> java Sfj  
Enter Number Of Process U want 2 Execute---  
4  
Enter arrival time of 1 Process :  
0  
Enter CPU time of 1 Process :  
8  
Enter arrival time of 2 Process :  
1  
Enter CPU time of 2 Process :  
4  
Enter arrival time of 3 Process :  
2  
Enter CPU time of 3 Process :  
9  
Enter arrival time of 4 Process :  
3  
Enter CPU time of 4 Process :  
5
```

Process	AT	CPU_T
2	1	4
4	3	5
1	0	8
3	2	9

```
Total turn around time is : 12.5  
Total waiting time is : 6.0  
PS D:\engineering\TE\spos> █
```

7)

```
import java.util.Scanner;
public class Bankers{
    private int need[],allocate[],max[],avail[],np,nr;

    private void input(){
        Scanner sc=new Scanner(System.in);
        System.out.print("Enter no. of processes and resources : ");
        np=sc.nextInt(); //no. of process
        nr=sc.nextInt(); //no. of resources
        need=new int[np][nr]; //initializing arrays
        max=new int[np][nr];
        allocate=new int[np][nr];
        avail=new int[1][nr];

        System.out.println("Enter allocation matrix -->");
        for(int i=0;i<np;i++)
            for(int j=0;j<nr;j++)
                allocate[i][j]=sc.nextInt(); //allocation matrix

        System.out.println("Enter max matrix -->");
        for(int i=0;i<np;i++)
            for(int j=0;j<nr;j++)
                max[i][j]=sc.nextInt(); //max matrix

        System.out.println("Enter available matrix -->");
        for(int j=0;j<nr;j++)
            avail[0][j]=sc.nextInt(); //available matrix

        sc.close();
    }

    private int[] calc_need(){
        for(int i=0;i<np;i++)
            for(int j=0;j<nr;j++) //calculating need matrix
                need[i][j]=max[i][j]-allocate[i][j];

        return need;
    }
}
```

```

private boolean check(int i){
    //checking if all resources for ith process can be allocated
    for(int j=0;j<nr;j++)
        if(avail[0][j]<need[i][j])
            return false;

    return true;
}

public void isSafe(){
    input();
    calc_need();
    boolean done[]=new boolean[np];
    int j=0;

    while(j<np){ //until all process allocated
        boolean allocated=false;
        for(int i=0;i<np;i++)
            if(!done[i] && check(i)){ //trying to allocate
                for(int k=0;k<nr;k++)
                    avail[0][k]=avail[0][k]-need[i][k]+max[i][k];
                System.out.println("Allocated process : "+i);
                allocated=done[i]=true;
                j++;
            }
        if(!allocated) break; //if no allocation
    }
    if(j==np) //if all processes are allocated
        System.out.println("\nSafely allocated");
    else
        System.out.println("All proceess cant be allocated safely");
}

public static void main(String[] args) {
    new Bankers().isSafe();
}
}

```

```
PS D:\engineering\TE\spos> javac Bankers.java
PS D:\engineering\TE\spos> java Bankers
Enter no. of processes and resources : 3 4
Enter allocation matrix -->
1 2 2 1
1 0 3 3
1 2 1 0
Enter max matrix -->
3 3 2 2
1 1 3 4
1 3 5 0
Enter available matrix -->
3 1 1 2
Allocated process : 0
Allocated process : 1
Allocated process : 2

Safely allocated
PS D:\engineering\TE\spos> █
```

## 6)Write a Java Program (using OOP features) to implement paging simulation using

### 1. FIFO

### 2. Least Recently Used (LRU)

### 3. Optimal algorithm

#### 6.1

```
import java.io.*;

public class Fifo {
    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);
}
}

```

```

PS D:\engineering\TE\spos> javac Fifo.java
PS D:\engineering\TE\spos> java Fifo
Please enter the number of Frames:
3
Please enter the length of the Reference string:
12
Please enter the reference string:
1
2
3
4
1
2
5
1
2
3
4
5

    1   1   1   4   4   4   5   5   5   5   5   5
   -1   2   2   2   1   1   1   1   1   3   3   3
   -1  -1   3   3   3   2   2   2   2   2   4   4
The number of Hits: 3
Hit Ratio: 0.25
The number of Faults: 9
PS D:\engineering\TE\spos>

```

## 6.2)Least Recently Used

```

import java.util.*;
class LruAlgo
{
int p[],n,fr[],m,fs[],index,k,l,flag1=0,flag2=0,pf=0,frsize=3,i,j;
Scanner src=new Scanner(System.in);
void read()
{

System.out.println("Enter page table size");
n=src.nextInt();
p=new int[n];
System.out.println("Enter element in page table");
for(int i=0;i<n;i++)
p[i]=src.nextInt();

System.out.println("Enter page frame size");
m=src.nextInt();
fr=new int[m];
fs=new int[m];

```



```

}

void display()
{

System.out.println("\n");
for(i=0;i<m;i++)
{

if(fr[i]==-1)
System.out.println(" ");
else
System.out.println "["+fr[i]+"]");
}
}

void lru()
{
for(i=0;i<m;i++)
{
fr[i]=-1;
}
for(j=0;j<n;j++)
{
flag1=0;flag2=0;
for(i=0;i<m;i++)
{
if(fr[i]==p[j])
{
flag1=1;
flag2=1;
break;
}
}
if(flag1==0)
{
for(i=0;i<m;i++)
{
if(fr[i]==-1)
{
fr[i]=p[j];
flag2=1;
break;
}
}
}
if(flag2==0)
{
for(i=0;i<3;i++)
fs[i]=0;

```

```

for(k=j-1,l=1;l<=frsize-1;l++,k--)
{
for(i=0;i<3;i++)
{
if(fr[i]==p[k])
fs[i]=1;
}
}
for(i=0;i<3;i++)
{
if(fs[i]==0)
index=i;
}
fr[index]=p[j];
pf++;
}
System.out.print("Page : "+p[j]);
display();
}
System.out.println("\n no of page faults :"+pf);
}

public static void main(String args[])
{
LruAlgo a=new LruAlgo();
a.read();
a.lru();
a.display();
}
}

```

PS D:\engineering\TE\spos> javac LruAlgo.java

PS D:\engineering\TE\spos> java LruAlgo

Enter page table size

10

Enter element in page table

1

5

1

2

6

2

7

1

5

1

Enter page frame size

3

Page : 1

[1]

[]

[]

Page : 5

[1]

[5]

[]

Page : 1

[1]

[5]

[]

Page : 2

[1]

[5]

[2]

Page : 6

[1]

[6]

[2]

Page : 2

[1]

[6]

[2]

Page : 7

[7]

[6]

[2]

Page : 1

[7]

[1]

[2]

Page : 5

[7]

[1]

[5]

Page : 1

[7]

[1]

[5]

no of page faults :4

[7]

[1]

[5]

### 6.3) Optimal algorithm

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
public class OptimalReplacement {

    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;
        boolean isFull = false;
        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)
{
    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);
}
}

```

```
PS D:\engineering\TE\spos> javac OptimalReplacement.java
```

```
PS D:\engineering\TE\spos> java OptimalReplacement
```

```
Please enter the number of Frames:
```

```
3
```

```
Please enter the length of the Reference string:
```

```
20
```

```
Please enter the reference string:
```

```
1
```

```
2
```

```
3
```

```
2
```

```
1
```

```
5
```

```
2
```

```
1
```

```
6
```

```
2
```

```
5
```

```
6
```

```
3
```

```
1
```

```
3
```

```
6
```

```
1
```

```
2
```

```
4
```

```
3
```

```

1  1  1  1  1  1  1  1  6  6  6  6  6  6  6  6  2  4  4
-1  2  2  2  2  2  2  2  2  2  2  2  2  1  1  1  1  1  1
-1 -1  3  3  3  5  5  5  5  5  5  5  3  3  3  3  3  3  3

```

```
The number of Hits: 11
```

```
Hit Ratio: 0.55
```

```
The number of Faults: 9
```

```
PS D:\engineering\TE\spos> █
```

