1.FCFS

import java.util.\*;

import java.lang.\*;

class Fcfs

{

public static void main (String args[])

{

Scanner s=new Scanner(System.in);

int p;

float t1=0,t2=0;

System.out.println("Enter number of processor");

p=s.nextInt();

int bt[]=new int[p];

System.out.println("Enter burst time");

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

{

System.out.println("Enter burst time for p"+(i+1)+"=");

bt[i]=s.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\t burst time\t twaiting time\t turnaround 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,att;

System.out.println("average waiting time="+t1/p+"ms");

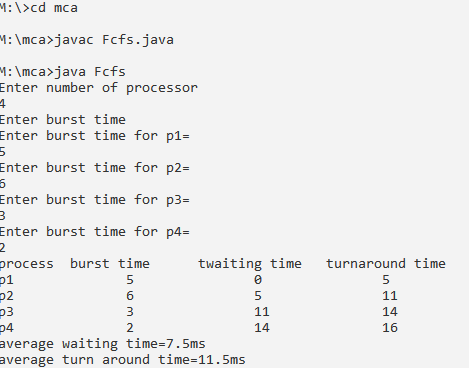
System.out.println("average turn around time="+t2/p+"ms");

}

}

}

**OUTPUT:**



2.SFJS

import java.util.\*;

import java.util.\*;

import java.lang.\*;

class Sjfs

{

public static void main(String args [])

{

Scanner s=new Scanner(System.in);

int p,temp,i,j,ord[];

float t1=0,t2=0;

System.out.println("enter number of processes");

p=s.nextInt();

int bt[]=new int[p];

ord=new int[p];

System.out.println("Enter burst time");

for(i=0;i<p;i++)

{

ord[i]=i+1;

bt[i]=s.nextInt();

}

for(i=0;i<p;i++)

{

for(j=i+1;j<p;j++)

{

if(bt[i]>bt[j])

{

temp=bt[i];

bt[i]=bt[j];

bt[j]=temp;

temp=ord[i];

ord[i] = ord[j];

ord[j]=temp;

}

}

}

int wt[]=new int[p];

wt[0]=0;

for(i=1;i<p;i++)

{

wt[i]=bt[i-1]+wt[i-1];

t1+=wt[i];

}

int tat[]=new int[p];

for(i=0;i<p;i++)

{

tat[i]=bt[i]+wt[i];

t2+=tat[i];

}

System.out.println("process \t burst time \t twaiting time \t turnaround time");

for(i=0;i<p;i++)

{

System.out.println("p"+ord[i]+"\t\t" +bt[i]+"\t\t"+wt[i]+"\t\t"+tat[i]);

}

float awt,att;

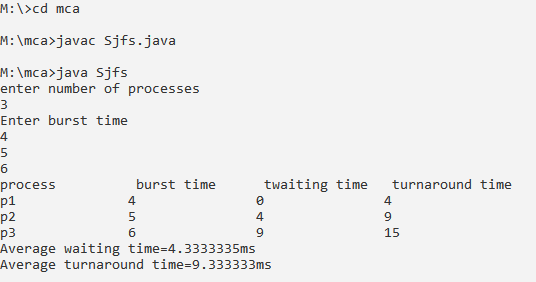
System.out.println("Average waiting time="+t1/p+"ms");

System.out.println("Average turnaround time="+t2/p+"ms");

}

}

OUTPUT**:**



3.PRIORITY

import java.util.\*;

import java.lang.\*;

class Priority

{

public static void main(String args[])

{

Scanner s=new Scanner(System.in);

int p,temp,i,j,order[];

float t1=0,t2=0;

System.out.println("enter no of processes");

p=s.nextInt();

int bt[]=new int[p];

order=new int[p];

int pr[]=new int[p];

System.out.println("enter burst time");

for(i=0;i<p;i++)

{

order[i]=i+1;

bt[i]=i+1;

}

System.out.println("enter priority");

for(i=0;i<p;i++)

{

order[i]=i+1;

pr[i]=s.nextInt();

}

for(i=0;i<p;i++)

{

for(j=i+1;j<p;j++)

{

if(pr[i]>pr[j])

{

temp=pr[i];

pr[j]=pr[j];

temp=bt[i];

bt[i]=bt[j];

bt[j]=temp;

temp=order[i];

order[i]=order[j];

order[j]=temp;

}

}

}

int wt[]=new int[p];

wt[0]=0;

for(i=1;i<p;i++)

{

wt[i]=bt[i-1]+wt[i-1];

t1+=wt[i];

}

int tat[]=new int[p];

for(i=0;i<p;i++)

{

tat[i]=bt[i]+wt[i];

t2+=tat[i];

}

System.out.println("process\t burst time\t priority\t waiting time \t turnaround time");

for(i=0;i<p;i++)

{

System.out.println("p"+order[i]+"\t\t"+bt[i]+"\t\t"+pr[i]+"\t\t"+wt[i]+"\t\t"+tat[i]);

}

float awt,att;

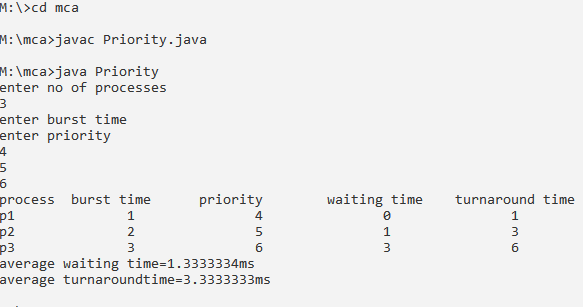
System.out.println("average waiting time="+t1/p+"ms");

System.out.println("average turnaroundtime="+t2/p+"ms");

}

}

OUTPUT:



4.ROUND ROBIN

import java.util.\*;

import java.lang.\*;

class RoundRobin

{

Scanner sc=new Scanner(System.in);

int[] bur,rem,wai,ta;

int size,q,b=0,t=0,flag=0;

RoundRobin(int size)

{

this.size=size;

bur=new int[size];

wai=new int[size];

ta=new int[size];

rem=new int[size];

}

void get()

{

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

{

System.out.println("enter burst time of p"+(i+1)+":");

bur[i]=rem[i]=sc.nextInt();

}

System.out.println("enter quantum time:");

q=sc.nextInt();

}

void round()

{

do

{

flag=0;

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

{

if(rem[i]>=q)

{

System.out.println("p"+(i+1)+"->");

for(int j=0;j<size;j++)

{

if(j==i)

rem[i]=rem[i]-q;

else if(rem[j]>0)

wai[j]+=q;

}

}

else if(rem[i]>0)

{

System.out.println("p"+(i+1)+"\t");

for(int j=0;j<size;j++)

if(j==i)

rem[i]=0;

else if(rem[j]>0)

wai[j]+=rem[i];

}

}

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

if(rem[i]>0)

flag=1;

}

while(flag==1);

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

ta[i]=wai[i]+bur[i];

}

void display()

{

System.out.println("\n process \tburst\twaiting\tturnaround");

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

{

System.out.println("\t p"+(i+1)+"\t"+bur[i]+"\t"+wai[i]+"\t"+ta[i]);

b=wai[i];

t+=ta[i];

}

System.out.println("Average waiting time:"+(b/size));

System.out.println("Average turnaround time:"+(t/size));

}

}

class KRoundRobin

{

public static void main(String args[])

{

Scanner s=new Scanner(System.in);

System.out.println("Enter no of process:");

int n=s.nextInt();

RoundRobin obj=new RoundRobin(n);

obj.get();

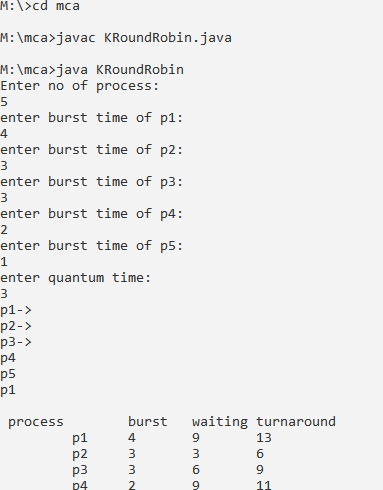
obj.round();

obj.display();

}

}

**OUTPUT:**





5.BANKERS ALGORITHM

import java.util.\*;

public class Bankers

{

private int need[][],allocate[][],max[][],avail[][],nr,np;

private void input()

{

Scanner sc=new Scanner(System.in);

System.out.println("Enter number of processses and resources");

np=sc.nextInt();

nr=sc.nextInt();

need=new int[np][nr];

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();

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();

System.out.println("Enter available matrix-->");

for(int j=0;j<nr;j++)

avail[0][j]=sc.nextInt();

sc.close();

}

private int[][] calc\_need()

{

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

for(int j=0;j<nr;j++)

need[i][j]=max[i][j]-allocate[i][j];

return need;

}

private boolean check(int i)

{

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)

{

boolean allocated=false;

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

if(!done[i]&&check(i))

{

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(j==np)

System.out.println("\n safely allocated");

else

System.out.println(" all process cant be allocated safely");

}

public static void main(String arg[])

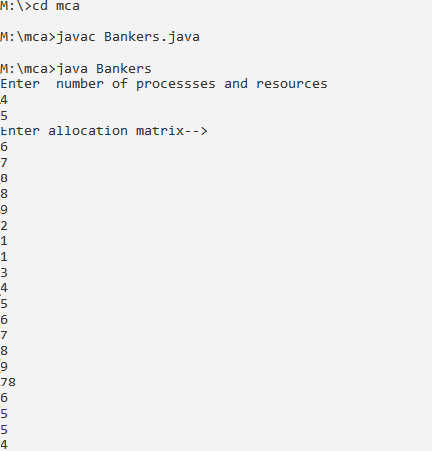
{

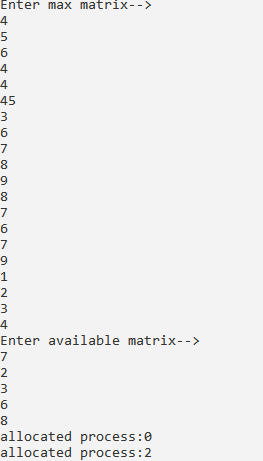
new Bankers().isSafe();

}

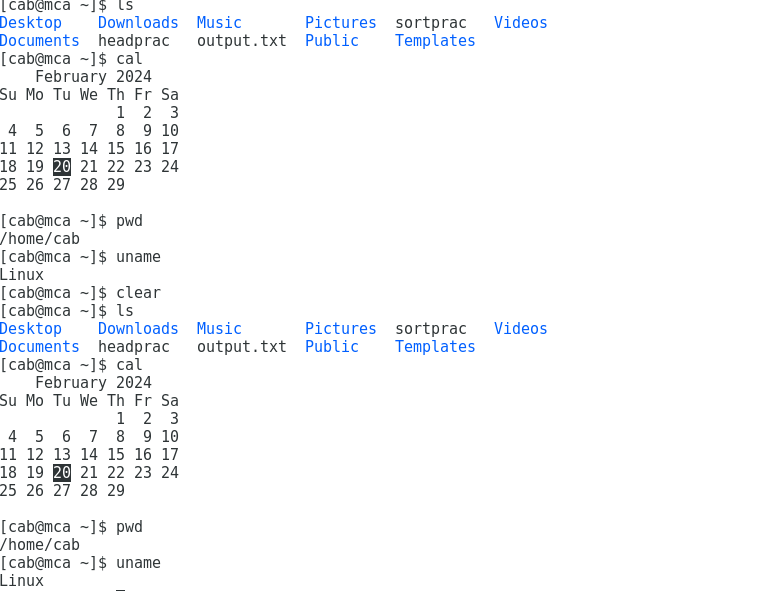
}

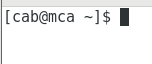
OUTPUT:





LINUX COMMANDS





12.LINUX COMMANDS

