**TASK # 1:**

Implement the algorithm FCFS.

SOURCE CODE:

import java.util.Scanner;

public class FCFS {

public static void main(String args[]){

Scanner sc = new Scanner(System.in);

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

int n = sc.nextInt();

int pid[] = new int[n];

int ar[] = new int[n];

int bt[] = new int[n];

int ct[] = new int[n];

int ta[] = new int[n];

int wt[] = new int[n];

int temp;

float avgwt=0,avgta=0;

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

{

System.out.println("enter process " + (i+1) + " arrival time: ");

ar[i] = sc.nextInt();

System.out.println("enter process " + (i+1) + " brust time: ");

bt[i] = sc.nextInt();

pid[i] = i+1; }

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

{

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

{

if( ar[j] > ar[j+1] )

{

temp = ar[j];

ar[j] = ar[j+1];

ar[j+1] = temp;

temp = bt[j];

bt[j] = bt[j+1];

bt[j+1] = temp;

temp = pid[j];

pid[j] = pid[j+1];

pid[j+1] = temp;

}} }

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

{

if( i == 0)

{

ct[i] = ar[i] + bt[i];

}

else

{

if( ar[i] > ct[i-1])

{

ct[i] = ar[i] + bt[i];

}

else

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

}

ta[i] = ct[i] - ar[i] ;

wt[i] = ta[i] - bt[i] ;

avgwt += wt[i] ;

avgta += ta[i] ;

}

System.out.println("\npid arrival brust complete turn waiting");

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

{

System.out.println(pid[i] + " \t " + ar[i] + "\t" + bt[i] + "\t" + ct[i] + "\t" + ta[i] + "\t" + wt[i] ) ;

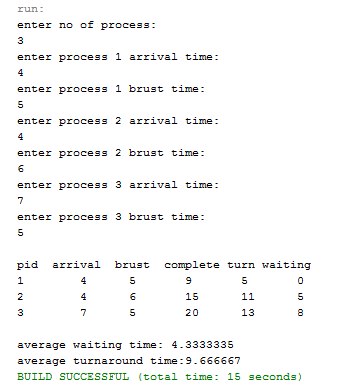
}

sc.close();

System.out.println("\naverage waiting time: "+ (avgwt/n));

System.out.println("average turnaround time:"+(avgta/n));}}

OUTPUT:



**TASK # 2:**

Implement the algorithm SJF nonpreemptive

SOURCE CODE:

import java.util.Scanner;

public class SJFnonprem {

public static void main(String args[])

{

Scanner sc = new Scanner(System.in);

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

int n = sc.nextInt();

int pid[] = new int[n];

int at[] = new int[n];

int bt[] = new int[n];

int ct[] = new int[n];

int ta[] = new int[n];

int wt[] = new int[n];

int f[] = new int[n];

int st=0, tot=0;

float avgwt=0, avgta=0;

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

{

System.out.println ("enter process " + (i+1) + " arrival time:");

at[i] = sc.nextInt();

System.out.println ("enter process " + (i+1) + " brust time:");

bt[i] = sc.nextInt();

pid[i] = i+1;

f[i] = 0;

}

boolean a = true;

while(true)

{

int c=n, min=999;

if (tot == n)

break;

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

{

if ((at[i] <= st) && (f[i] == 0) && (bt[i]<min))

{

min=bt[i];

c=i;

}}

if (c==n)

st++;

else

{

ct[c]=st+bt[c];

st+=bt[c];

ta[c]=ct[c]-at[c];

wt[c]=ta[c]-bt[c];

f[c]=1;

tot++;

}}

System.out.println("\npid arrival brust complete turn waiting");

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

{

avgwt+= wt[i];

avgta+= ta[i];

System.out.println(pid[i]+"\t"+at[i]+"\t"+bt[i]+"\t"+ct[i]+"\t"+ta[i]+"\t"+wt[i]);

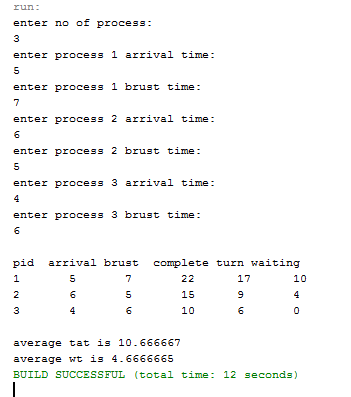
}

System.out.println ("\naverage tat is "+ (float)(avgta/n));

System.out.println ("average wt is "+ (float)(avgwt/n));

sc.close();}}

OUTPUT:



**TASK # 3:**

Implement the algorithm SJF preemptive

SOURCE CODE:

import java.util.Scanner;

public class SJFpreem {

public static void main (String args[])

{

Scanner sc=new Scanner(System.in);

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

int n= sc.nextInt();

int pid[] = new int[n];

int at[] = new int[n];

int bt[] = new int[n];

int ct[] = new int[n];

int ta[] = new int[n];

int wt[] = new int[n];

int f[] = new int[n];

int k[]= new int[n];

int i, st=0, tot=0;

float avgwt=0, avgta=0;

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

{

pid[i]= i+1;

System.out.println ("enter process " +(i+1)+ " arrival time:");

at[i]= sc.nextInt();

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

bt[i]= sc.nextInt();

k[i]= bt[i];

f[i]= 0;

}

while(true){

int min=99,c=n;

if (tot==n)

break;

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

{

if ((at[i]<=st) && (f[i]==0) && (bt[i]<min))

{

min=bt[i];

c=i;

}

}

if (c==n)

st++;

else

{

bt[c]--;

st++;

if (bt[c]==0)

{

ct[c]= st;

f[c]=1;

tot++;

}}}

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

{

ta[i] = ct[i] - at[i];

wt[i] = ta[i] - k[i];

avgwt+= wt[i];

avgta+= ta[i];

}

System.out.println("pid arrival burst complete turn waiting");

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

{

System.out.println(pid[i] +"\t"+ at[i]+"\t"+ k[i] +"\t"+ ct[i] +"\t"+ ta[i] +"\t"+ wt[i]);

}

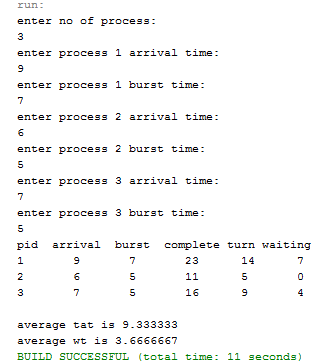
System.out.println("\naverage tat is "+ (float)(avgta/n));

System.out.println("average wt is "+ (float)(avgwt/n));

sc.close();

}}

OUTPUT:



**TASK # 4:**

Implement the algorithm priority

SOURCE CODE:

#include<stdio.h>

int main()

{

int bt[20],p[20],wt[20],tat[20],pr[20],i,j,n,total=0,pos,temp,avg\_wt,avg\_tat;

printf("Enter Total Number of Process:");

scanf("%d",&n);

printf("\nEnter Burst Time and Priority\n");

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

{

printf("\nP[%d]\n",i+1);

printf("Burst Time:");

scanf("%d",&bt[i]);

printf("Priority:");

scanf("%d",&pr[i]);

p[i]=i+1;

}

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

{

pos=i;

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

{

if(pr[j]<pr[pos])

pos=j;

}

temp=pr[i];

pr[i]=pr[pos];

pr[pos]=temp;

temp=bt[i];

bt[i]=bt[pos];

bt[pos]=temp;

temp=p[i];

p[i]=p[pos];

p[pos]=temp;

}

wt[0]=0;

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

{

wt[i]=0;

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

wt[i]+=bt[j];

total+=wt[i];

}

avg\_wt=total/n;

total=0;

printf("\nProcess\t    Burst Time    \tWaiting Time\tTurnaround Time");

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

{

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

total+=tat[i];

printf("\nP[%d]\t\t  %d\t\t    %d\t\t\t%d",p[i],bt[i],wt[i],tat[i]);

}

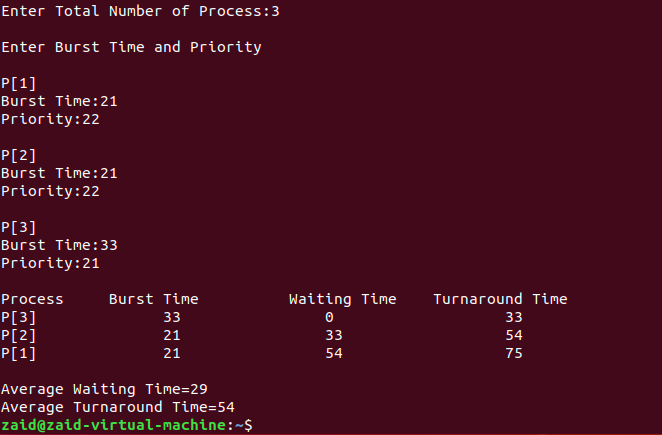
avg\_tat=total/n;

printf("\n\nAverage Waiting Time=%d",avg\_wt);

printf("\nAverage Turnaround Time=%d\n",avg\_tat);

}

OUTPUT:



**TASK # 5:**

Implement the algorithm Round Robin

SOURCE CODE:

include<stdio.h>

int main()

{

int count,j,n,time,remain,flag=0,time\_quantum;

int wait\_time=0,turnaround\_time=0,at[10],bt[10],rt[10];

printf("Enter Total Process:\t ");

scanf("%d",&n);

remain=n;

for(count=0;count<n;count++)

{

printf("Enter Arrival Time and Burst Time for Process Process Number %d :",count+1);

scanf("%d",&at[count]);

scanf("%d",&bt[count]);

rt[count]=bt[count];

}

printf("Enter Time Quantum:\t");

scanf("%d",&time\_quantum);

printf("\n\nProcess\t|Turnaround Time|Waiting Time\n\n");

for(time=0,count=0;remain!=0;)

{

if(rt[count]<=time\_quantum && rt[count]>0)

{

time+=rt[count];

rt[count]=0;

flag=1;

}

else if(rt[count]>0)

{

rt[count]-=time\_quantum;

time+=time\_quantum;

}

if(rt[count]==0 && flag==1)

{

remain--;

printf("P[%d]\t|\t%d\t|\t%d\n",count+1,time-at[count],time-at[count]-bt[count]);

wait\_time+=time-at[count]-bt[count];

turnaround\_time+=time-at[count];

flag=0;

} if(count==n-1)

count=0;

else if(at[count+1]<=time)

count++;

else

count=0;

}

printf("\nAverage Waiting Time= %f\n",wait\_time\*1.0/n);

printf("Avg Turnaround Time = %f",turnaround\_time\*1.0/n);

}

OUTPUT:

