# CODE

# #include<stdio.h>

# struct proc

# {

# int pid;

# int at,bt,wt,tat,rbt;

# int flag,flag1;

# };

# struct proc p1[10];

# int i,j,k,n,no,m;

# float atat=0.0,awt=0.0;

# int tbt=0;

# int minimum1();

# int main()

# {

# int minv,locv,mins,locs;

# printf("\nenter the number of processes:");

# scanf("%d",&n);

# printf("\nenter the proc information:");

# printf("\npid at bt");

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

# {

# p1[i].wt=0;

# p1[i].tat=0;

# p1[i].flag=0;

# p1[i].flag1=0;

# scanf("%d%d%d",&p1[i].pid,&p1[i].at,&p1[i].bt);

# tbt+=p1[i].bt;

# p1[i].rbt=p1[i].bt;

# }

# printf("\nthe proc information:");

# printf("\npid at bt");

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

# {

# printf("\n%d %d %d",p1[i].pid,p1[i].at,p1[i].bt);

# }

# minv=p1[0].at;

# locv=0;

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

# {

# if(p1[i].at<minv)

# {

# locv=i; //tells min at process in locv

# minv=p1[i].at;

# }

# }

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

# {

# if(p1[i].at==minv)

# {

# p1[i].flag1=1; //processes having same minimum at

# }

# }

# mins=p1[0].bt;

# locs=0;

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

# {

# if(p1[i].flag1==1&&p1[i].bt<mins)

# {

# mins=p1[i].bt; //gives process with minimum burst time

# locs=i;

# }

# }

# printf("\ngantt chart:");

# for(i=minv;i<tbt+minv;i++)

# {

# for(j=0;j<n;j++)

# {

# if(p1[j].rbt>0&&p1[j].at<=i)

# {

# p1[j].flag=1;

# }

# }

# no=minimum1();

# printf("%d p[%d]",i,p1[no].pid);

# p1[no].rbt=p1[no].rbt-1;

# for(k=0;k<n;k++)

# {

# if(p1[k].rbt>0&&p1[k].at<=i&&k!=no)

# {

# p1[k].wt++;

# }

# }

# }

# printf("%d",tbt+minv);

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

# {

# awt+=p1[i].wt;

# }

# awt=awt/n;

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

# {

# p1[i].tat=p1[i].wt+p1[i].bt;

# atat+=p1[i].tat;

# }

# atat=atat/n;

# printf("\n average wt=%f, average tat=%f",awt,atat);

# printf("\nthe proc information:");

# printf("\npid at bt wt tat");

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

# {

# printf("\n%d %d %d %d %d",p1[i].pid,p1[i].at,p1[i].bt,p1[i].wt,p1[i].tat);

# }

# }

# int minimum1()

# {

# int loc,z;

# int mini;

# mini=99;

# loc=-1;

# for(z=0;z<n;z++)

# {

# if(p1[z].rbt>0&&p1[z].at<=i&&p1[z].rbt<mini)

# {

# mini=p1[z].rbt;

# loc=z;

# }

# }

# return loc;

# }

# TEST CASE:

# #include<stdio.h>

# main()

# {

# int n,i,j,avgw=0;

# float avgt=0,temp;

# int burst[10],wait[10],process[10];

# printf("\nProcess Scheduling(SJFS) :\n");

# printf("\nEnter the number of process : ");

# 

# scanf("%d",&n);

# 

# printf("Enter the Burst time\n");

# 

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

# {

# printf("Burst time P %d : ",i+1);

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

# process[i] = i;

# printf("\n");

# }

# 

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

# {

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

# {

# if(burst[i]>burst[j])

# {

# temp = burst[i];

# burst[i] = burst[j];

# burst[j] = temp;

# temp = process[i];

# process[i] = process[j];

# process[j] = temp;

# }

# }

# }

# printf("\nWaiting time | Turn around time \n");

# 

# j=0;

# 

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

# {

# wait[i] = j;

# j = burst[i] + j;

# printf("\nP%d : %d\t\t%d",process[i]+1,wait[i],wait[i]+burst[i]);

# avgw+=wait[i];

# avgt+=wait[i]+burst[i];

# }

# printf("\n\n");

# printf("Average waiting time : %f\nAverage Turn around time : %f\n\n",avgw/(float)n,avgt/(float)n);

# }