#include <stdio.h>

struct process

{

    int WT,AT,BT,TAT,PT;

};

struct process a[10];

//Function to swap two variables

void swap(int \*a,int \*b)

{

    int temp=\*a;

    \*a=\*b;

    \*b=temp;

}

// For Non Preemptive

int Non\_preemptive()

{

    int n;

    float total\_wt=0,total\_tat=0,avg\_wt,avg\_tat;

    printf("Enter Number of Processes: ");

    scanf("%d",&n);

    printf("\n\t!!!!Higher number -> Higher priority!!!!\n");

    // b is array for burst time, p for priority and index for process id

    int b[n],p[n],index[n];

    for(int i=0;i<n;i++)

    {

        printf("Enter Burst Time and Priority Value for Process %d: ",i+1);

        scanf("%d %d",&b[i],&p[i]);

        index[i]=i+1;

    }

    for(int i=0;i<n;i++)

    {

        int a=p[i],m=i;

        //Finding out highest priority element and placing it at its desired position

        for(int j=i;j<n;j++)

        {

            if(p[j] > a)

            {

                a=p[j];

                m=j;

            }

        }

        //Swapping processes

        swap(&p[i], &p[m]);

        swap(&b[i], &b[m]);

        swap(&index[i],&index[m]);

    }

    // T stores the starting time of process

    int t=0;

    //Printing scheduled process

    printf("Order of process Execution is\n");

    for(int i=0;i<n;i++)

    {

        printf("P%d is executed from %d to %d\n",index[i],t,t+b[i]);

        t+=b[i];

    }

    printf("\n");

    printf("Process Id \tBurst Time\t Wait Time \tTurnAround Time\n");

    int wait\_time=0;

    for(int i=0;i<n;i++)

    {

        printf("P%d\t\t %d\t\t %d\t\t %d\n",index[i],b[i],wait\_time,wait\_time + b[i]);

        total\_wt = total\_wt + wait\_time;

        wait\_time += b[i];

        total\_tat = total\_tat +wait\_time;

    }

    avg\_wt= total\_wt/n;

    avg\_tat = total\_tat/n;

    printf("Average waiting time:%f\n",avg\_wt);

    printf("Average TrunAround time:%f\n",avg\_tat);

    return 0;

}

    // For Preemptive

int preemptive()

{

    int n,temp[10],t,count=0,short\_p;

    float total\_WT=0,total\_TAT=0,Avg\_WT,Avg\_TAT;

    printf("Enter the number of the process\n");

    scanf("%d",&n);

    printf("Enter the arrival time , burst time and priority of the process\n");

    printf("AT BT PT\n");

    for(int i=0;i<n;i++)

    {

        scanf("%d%d%d",&a[i].AT,&a[i].BT,&a[i].PT);

        // copying the burst time in

        // a temp array fot futher use

        temp[i]=a[i].BT;

    }

    // we initialize the burst time

    // of a process with maximum

    a[9].PT=10000;

    for(t=0;count!=n;t++)

    {

        short\_p=9;

        for(int i=0;i<n;i++)

        {

            if(a[short\_p].PT>a[i].PT && a[i].AT<=t && a[i].BT>0)

            {

                short\_p=i;

            }

        }

        a[short\_p].BT=a[short\_p].BT-1;

        // if any process is completed

        if(a[short\_p].BT==0)

        {

            // one process is completed

            // so count increases by 1

            count++;

            a[short\_p].WT=t+1-a[short\_p].AT-temp[short\_p];

            a[short\_p].TAT=t+1-a[short\_p].AT;

            // total calculation

            total\_WT=total\_WT+a[short\_p].WT;

            total\_TAT=total\_TAT+a[short\_p].TAT;

        }

    }

    Avg\_WT=total\_WT/n;

    Avg\_TAT=total\_TAT/n;

    // printing of the answer

    printf("ID WT TAT\n");

    for(int i=0;i<n;i++)

    {

        printf("%d %d\t%d\n",i+1,a[i].WT,a[i].TAT);

    }

    printf("Avg waiting time of the process  is %f\n",Avg\_WT);

    printf("Avg turn around time of the process is %f\n",Avg\_TAT);

    return 0;

}

int main(){

    int choice;

    printf("Enter the scheduling type:\n1. Preemptive\n2. Non Preemptive\t:");

    scanf("%d",&choice);

    if (choice==1)

    {

        printf("\t\*\*\*\*\*\*PREEMPTIVE PRIORITY SCHEDULING\*\*\*\*\*\*\n");

        preemptive();

    }else if (choice==2)

    {

        printf("\t\*\*\*\*\*\*NON-PREEMPTIVE PRIORITY SCHEDULING\*\*\*\*\*\*\n");

        Non\_preemptive();

    }else{

        printf("\t\tInvalid choice!\n\t\tExit Terminal\n");

    }

    return 0;

}