#include<stdio.h>

#include<stdlib.h>

//Symbol for complexity: n = Total no. of process

// m = Turn Around Time

int main(){

int noOfProcess;

int i, at, pr, bt;

int t=0;

int complete=0;

int minPriIndexC;

int minPri, minPriIndex;

//Taking User Input

printf("No of process: ");

scanf("%d",&noOfProcess);

int arr[8][noOfProcess];

for(i=0; i<noOfProcess; i++){

printf("\nFor Process P%d \n", i);

printf("Priority: ");

scanf("%d", &pr);

printf("Arrival Time: ");

scanf("%d", &at);

printf("Burst Time: ");

scanf("%d", &bt);

arr[0][i]=i;

arr[1][i]=at; //Arrival Time

arr[2][i]=pr; //Priority

arr[3][i]=bt; //Burst Time

arr[4][i]=0; //Wait Time

arr[5][i]=0; //State

arr[6][i]=0; //Turn Around Time

arr[7][i]=0; //WaitTime to incriment Priority

}

printf("\n-----------------------------------\n\n");

//Saving the burst time of each process.

int burstTime[noOfProcess];

for(i=0; i<noOfProcess;i++){

burstTime[i]=arr[3][i];

}

int initialPri[noOfProcess];

for(i=0; i<noOfProcess;i++){

initialPri[i]=arr[2][i];

}

//Checking initial minimum priority.

for(i=0; i<noOfProcess; i++){

if(arr[1][i]==0){

minPriIndexC = i;

break;

}

}

while(complete==0){

//Checking for the arrival time

for(i=0;i<noOfProcess; i++){

if(arr[1][i]==t){

arr[5][i]=2;

}

}

minPri = 100000;

minPriIndex = 0;

//Finding the minimum priority

for(i=0; i<noOfProcess; i++){

if(arr[2][i]<minPri&&arr[5][i]==2){

minPriIndex = i;

minPri = arr[2][i];

}

}

t++;

if(minPriIndexC!=minPriIndex){

int time=2;

while(time!=0){

//Checking for the arrival time

for(i=0;i<noOfProcess; i++){

if(arr[1][i]==t){

arr[5][i]=2;

}

}

t++;

for(i=0; i<noOfProcess;i++){

if(arr[5][i]==2){

arr[4][i] +=1;

arr[7][i] +=1;

if(arr[7][i]==2){

arr[7][i]=0;

arr[2][i] -=1;

}

}

}

time--;

}

}

//Execution

minPriIndexC = minPriIndex;

arr[3][minPriIndex] -=1;

arr[4][minPriIndex] -=1;

arr[7][minPriIndex] -=1;

if(arr[3][minPriIndex]==0){

arr[6][minPriIndex] = t; //--

arr[5][minPriIndex] = -1;

arr[7][minPriIndex] = 0; //--

}

//Checking for the completed process.

for(i=0; i<noOfProcess;i++){

if(arr[5][i]==2){

arr[4][i] +=1;

arr[7][i] +=1;

if(arr[7][i]==2){

arr[7][i]=0;

arr[2][i] -=1;

}

}

}

//Checking the completion

int temp=0;

for(i=0; i<noOfProcess;i++){

temp=temp+arr[5][i];

}

if((temp\*(-1))==noOfProcess){

complete=-1;

}

}

/\*End of the logic\*/

//Output

printf("Priority Scheduling Table: \n");

printf("\nProcNo: \t");

for(i=0; i<noOfProcess; i++){

printf("P%d\t", arr[0][i]);

}

printf("\nArrT: \t");

for(i=0; i<noOfProcess; i++){

printf("%d\t", arr[1][i]);

}

printf("\nInitPri: \t"); for(i=0; i<noOfProcess; i++){

printf("%d\t", initialPri[i]);

}

printf("\nBurstT: \t");

for(i=0; i<noOfProcess; i++){

printf("%d\t", burstTime[i]);

}

printf("\nWaitT: \t"); for(i=0; i<noOfProcess; i++){

printf("%d\t", (arr[6][i]-arr[1][i])-burstTime[i]); }

printf("\nCompT: \t");

for(i=0; i<noOfProcess; i++){

printf("%d\t", arr[6][i]);

}

printf("\nTAT: \t");

for(i=0; i<noOfProcess; i++){

printf("%d\t", arr[6][i]-arr[1][i]); }

printf("\nFinalPri: \t");

for(i=0; i<noOfProcess; i++){

printf("%d\t", arr[2][i]);

}

printf("\n\n\nRequired Answers: ");

printf("\n\nPNo: |\t");

for(i=0; i<noOfProcess; i++){

printf(" P%d", arr[0][i]);

printf(" | ");

}

printf("\n\nTAT: |\t");

for(i=0; i<noOfProcess; i++){

printf("%5d", arr[6][i]-arr[1][i]);

printf(" | ");

}

printf("\n\nWT: |\t");

int totalWaitTime=0;

for(i=0; i<noOfProcess; i++){

printf("%5d", (arr[6][i]-arr[1][i])-burstTime[i]);

totalWaitTime += (arr[6][i]-arr[1][i])-burstTime[i];

printf(" | ");

}

printf("\n\nTurn Around Time: %d second", t);

printf("\n\nAverage Waiting Time : %.2f second\n\n", totalWaitTime/(float)noOfProcess );

}