**Problem 7:** Write a program to implement the preemptive priority scheduling algorithm and find the average turnaround time, waiting time, completion time and response time for overall process. Also Print Gantt chart for it.

**Solution:**

*Source code:*

#include<iostream>

#include<algorithm>

using namespace std;

struct node{

char process\_name;

int burst\_time;

int arrival\_time;

int response\_time;

int priority;

int wait\_time;

int complete\_time;

}arr[1000],brr[1000],crr[1000];

void insert(int n){

int i;

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

cin>>arr[i].process\_name;

cin>>arr[i].priority;

cin>>arr[i].arrival\_time;

cin>>arr[i].burst\_time;

arr[i].wait\_time=-arr[i].arrival\_time+1;

}

}

bool arrival\_time\_sort(node arr,node brr){

return arr.arrival\_time < brr.arrival\_time;

}

bool prioritySort(node arr,node brr){

return arr.priority < brr.priority;

}

int k=0,f=0,r=0;

void disp(int nop,int qt){

int n=nop,q;

sort(arr,arr+n,arrival\_time\_sort);

int ttime=0,i;

int j,tArray[n];

int alltime=0;

bool moveLast=false;

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

alltime+=arr[i].burst\_time;

}

alltime+=arr[0].arrival\_time;

for(i=0;ttime<=alltime;){

j=i;

while(arr[j].arrival\_time<=ttime&&j!=n){

brr[r]=arr[j];

j++;

r++;

}

if(r==f){

crr[k].process\_name='i';

crr[k].burst\_time=arr[j].arrival\_time-ttime;

crr[k].arrival\_time=ttime;

ttime+=crr[k].burst\_time;

k++;

continue;

}

i=j;

if(moveLast==true){

sort(brr+f,brr+r,prioritySort);

}

j=f;

if(brr[j].burst\_time>qt){

crr[k]=brr[j];

crr[k].burst\_time=qt;

k++;

brr[j].burst\_time=brr[j].burst\_time-qt;

ttime+=qt;

moveLast=true;

for(q=0;q<n;q++){

if(brr[j].process\_name!=arr[q].process\_name){

arr[q].wait\_time+=qt;

}

}

}

else{

crr[k]=brr[j];

k++;

f++;

ttime+=brr[j].burst\_time;

moveLast=false;

for(q=0;q<n;q++){

if(brr[j].process\_name!=arr[q].process\_name){

arr[q].wait\_time+=brr[j].burst\_time;

}

}

}

if(f==r&&i>=n)

break;

}

tArray[i]=ttime;

ttime+=arr[i].burst\_time;

for(i=0;i<k-1;i++){

if(crr[i].process\_name==crr[i+1].process\_name){

crr[i].burst\_time+=crr[i+1].burst\_time;

for(j=i+1;j<k-1;j++)

crr[j]=crr[j+1];

k--;

i--;

}

}

int rtime=0;

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

rtime=0;

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

if(crr[i].process\_name==arr[j].process\_name){

arr[j].response\_time=rtime;

break;

}

rtime+=crr[i].burst\_time;

}

}

float averageWaitingTime=0;

float averageResponseTime=0;

float averageTAT=0;

cout<<"\nGantt Chart\n";

rtime=0;

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

if(i!=k)

cout<<"| "<<'P'<< crr[i].process\_name << " ";

rtime+=crr[i].burst\_time;

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

if(arr[j].process\_name==crr[i].process\_name)

arr[j].complete\_time=rtime;

}

}

cout<<"|\n";

rtime=0;

for (i=0; i<k+1; i++){

cout << rtime << "\t";

tArray[i]=rtime;

rtime+=crr[i].burst\_time;

}

cout<<"\n";

cout<<"\n";

cout<<" Process Name| Priority| Arrival Time| Burst Time| Complete Time| Turn Around Time| Wait Time| Response Time|\n";

for (i=0; i<nop&&arr[i].process\_name!='i'; i++){

if(arr[i].process\_name=='\0')

break;

cout <<" P"<< arr[i].process\_name;

cout <<" "<<arr[i].priority;

cout <<" " <<arr[i].arrival\_time;

cout <<" "<< arr[i].burst\_time;

cout <<" "<< arr[i].complete\_time;

cout <<" "<<arr[i].wait\_time+arr[i].complete\_time-rtime+arr[i].burst\_time;

averageTAT+=arr[i].wait\_time+arr[i].complete\_time-rtime+arr[i].burst\_time;

cout <<" "<< arr[i].wait\_time+arr[i].complete\_time-rtime;

averageWaitingTime+=arr[i].wait\_time+arr[i].complete\_time-rtime;

cout <<" "<<arr[i].response\_time-arr[i].arrival\_time;

averageResponseTime+=arr[i].response\_time-arr[i].arrival\_time;

cout <<"\n";

}

cout<<"Average Waiting Time: "<<(float)averageWaitingTime/(float)n<<endl;

cout<<"Average Turn Around Time: "<<(float)averageTAT/(float)n<<endl;

}

int main(){

int nop,choice,i,qt;

cout<<"Enter number of processes\n";

cin>>nop;

cout<<"Enter Process Name, Priority, Arrival Time, Burst Time\n";

insert(nop);

disp(nop,1);

return 0;

}

*Output:*

