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:

```
D:\os lab\Tanmay-Vig_19BCS061_p7.exe
Enter number of processes
4
Enter Process Name, Priority, Arrival Time, Burst Time
1 4 0 8
2 1 2 6
3 3 5 10
4 2 6 11

Gantt Chart
|  P1   |  P2   |  P4   |  P3   |  P1   |
0       2       8       19      29      35

 Process Name| Priority| Arrival Time| Burst Time| Complete Time| Turn Around Time| Wait Time| Response Time|
          P1        4           0          8            35               35        27           0
          P2        1           2          6             8              6         0           0
          P3        3           5          10             29               24        14           14
          P4        2           6          11             19               13        2           2
Average Waiting Time: 10.75
Average Turn Around Time: 19.5

--------------------------------
Process exited after 31.2 seconds with return value 0
Press any key to continue . . .
```