Question 7

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

Garv nanwani

19BCS049

CODE :

```
#include<iostream>
#include<algorithm>
using namespace std;

struct node{
    char pname;
    int btime;
    int atime;
    int priority;
    int restime=0;
    int ctime=0;
    int wtime=0;
}a[1000],b[1000],c[1000];

void insert(int n){
    int i;
    for(i=0;i<n;i++){
        cin>>a[i].pname;
        cin>>a[i].priority;
        cin>>a[i].atime;
        cin>>a[i].btime;
        a[i].wtime=-a[i].atime+1;
    }
}

bool btimeSort(node a,node b){
    return a.btime < b.btime;
}

bool atimeSort(node a,node b){
    return a.atime < b.atime;
}
bool prioritySort(node a,node b){
    return a.priority < b.priority;
}
int k=0,f=0,r=0;
void disp(int nop,int qt){
    int n=nop,q;
    sort(a,a+n,atimeSort);
    int ttime=0,i;
    int j,tArray[n];
```

```
    int alltime=0;
    bool moveLast=false;
    for(i=0;i<n;i++){
        alltime+=a[i].btime;
    }
    alltime+=a[0].atime;
    for(i=0;ttime<=alltime;){
        j=i;
        while(a[j].atime<=ttime&&j!=n){
            b[r]=a[j];
            j++;
            r++;
        }
        if(r==f){
            c[k].pname='i';
            c[k].btime=a[j].atime-ttime;
            c[k].atime=ttime;
            ttime+=c[k].btime;
            k++;
            continue;
        }
        i=j;
        if(moveLast==true){
            sort(b+f,b+r,prioritySort);
        }

        j=f;
        if(b[j].btime>qt){
            c[k]=b[j];
            c[k].btime=qt;
            k++;
            b[j].btime=b[j].btime-qt;
            ttime+=qt;
            moveLast=true;
            for(q=0;q<n;q++){
                if(b[j].pname!=a[q].pname){
                    a[q].wtime+=qt;
                }
            }
        }
        else{
            c[k]=b[j];
            k++;
            f++;
            ttime+=b[j].btime;
            moveLast=false;
            for(q=0;q<n;q++){
                if(b[j].pname!=a[q].pname){
                    a[q].wtime+=b[j].btime;
                }
            }
        }
```

```
        if(f==r&&i>=n)
        break;
    }
    tArray[i]=ttime;
    ttime+=a[i].btime;
    for(i=0;i<k-1;i++){
        if(c[i].pname==c[i+1].pname){
            c[i].btime+=c[i+1].btime;
            for(j=i+1;j<k-1;j++)
                c[j]=c[j+1];
            k--;
            i--;
        }
    }

    int rtime=0;
    for(j=0;j<n;j++){
        rtime=0;
        for(i=0;i<k;i++){
            if(c[i].pname==a[j].pname){
                a[j].restime=rtime;
                break;
            }
            rtime+=c[i].btime;
        }
    }

    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'<< c[i].pname << "   ";
        rtime+=c[i].btime;
        for(j=0;j<n;j++){
            if(a[j].pname==c[i].pname)
                a[j].ctime=rtime;
        }
    }
    cout<<"\n";
    rtime=0;
    for (i=0; i<k+1; i++){
        cout << rtime << "\t";
        tArray[i]=rtime;
        rtime+=c[i].btime;
    }

    cout<<"\n";
    cout<<"\n";
```

```
    cout<<"P.Name Priority AT\tBT\tCT\tTAT\tWT\tRT\n";
    for (i=0; i<nop&&a[i].pname!='i'; i++){
        if(a[i].pname=='\0')
            break;
        cout <<'P'<< a[i].pname << "\t";
        cout << a[i].priority << "\t";
        cout << a[i].atime << "\t";
        cout << a[i].btime << "\t";
        cout << a[i].ctime << "\t";
        cout << a[i].wtime+a[i].ctime-rtime+a[i].btime << "\t";
        averageTAT+=a[i].wtime+a[i].ctime-rtime+a[i].btime;
        cout << a[i].wtime+a[i].ctime-rtime << "\t";
        averageWaitingTime+=a[i].wtime+a[i].ctime-rtime;
        cout << a[i].restime-a[i].atime << "\t";
        averageResponseTime+=a[i].restime-a[i].atime;
        cout <<"\n";
    }
    cout<<"Average Waiting time: "<<(float)averageWaitingTime/(float)n<<endl;
    cout<<"Average TA time: "<<(float)averageTAT/(float)n<<endl;

}

int main(){
    int nop,choice,i,qt;
    cout<<"Enter number of processes\n";
    cin>>nop;
    cout<<"Enter process, priority, AT, BT\n";
    insert(nop);
    disp(nop,1);
    return 0;
}
```

Output :

```
Enter number of processes
3
Enter process, priority, AT, BT
1 2 4 6
2 4 5 4
3 4 6 7

Gantt Chart
|  Pi   |  P1   |  P2   |  P3
0       4       10      14      21

P.Name Priority AT      BT      CT      TAT     WT      RT
P1      2       4       6       10      2       -4      0
P2      4       5       4       14      5       1       5
P3      4       6       7       21      11      4       8
Average Waiting time: 0.333333
Average TA time: 6
```