Ex. No.: 6c) 6
Date: 3.25

PRIORITY SCHEDULING

Aim:

To implement priority scheduling technique

Algorithm:

1. Get the number of processes from the user.
2. Read the process name, burst time and priority of process.
3. Sort based on burst time of all processes in ascending order based priority 4.
Calculate the total waiting time and total turnaround time for each process 5.
Display the process name & burst time for each process.
6. Display the total waiting time, average waiting time, turnaround time

Program Code:

```
#include <stdio.h>
int main()
{
    int n;
    Printf ("Enter the no. of processes : ");
     scanf ("%d", &n);
     int at[n], bt[n], priority [n], wt[n], tat[n], ct[n],
                                                    pro[n];

     for( int i=0; i<n; i++)
     {
          pro[i] = i+1;
          Printf ("Process %d Arrival Time : ", i+1);
          scanf ("%d", & at[i]);
          Printf ("Process %d BurstTime: ", i+1);
          Scanf ("%d", & bt[i]);
          Printf(" Process %d Priority : ", i+1);
          scanf("%d", & priority[i]);
     }
     for (int i=0 ; i <n-1; i++)
     { for (int j =0; j< n-i-1 ; j++)
          {
```

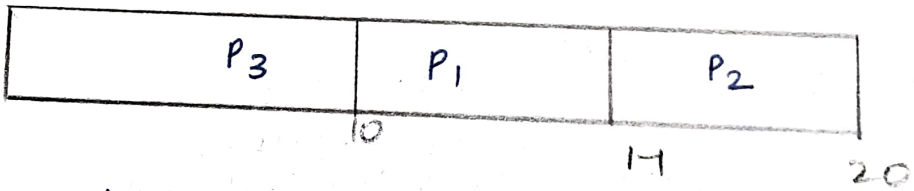
```
if (at[j] > at [j+1])
{
    int temp;
    temp = at [j];
    at[j] = at[j+1];
    at [j+1] = temp;
    temp = priority [j];
    priority[j] = priority [j+1];
    priority [j+1] = temp;
    temp = pro[j];
    pro[j] = pro[j+1];
    pro[j+1] = temp;
}
}
}
int time=0, comp=0; float total_tat=0; total_wt=0;
while (comp<n){
    int start = comp, end = comp;
    while (end<n && at [end] <= time)
        end ++;
    for ( int i=start ; i< end-1 ; i++ ){
        for ( int j =start; j< end -i-1 ; j++ ) {
            if (priority [j] > priority [j+1]){
                int temp;
                temp =at [j];
                at[j] =at[j+1];
                at[j+1] =temp;
                temp = bt [j]; bt[j] = bt[j+1];
                bt[j+1] = temp;
                temp = priority [j]; priority[j] =
                                    priority[j+1];
                Priority[j+1] = temp;
                temp = pro [j]; pro [j] =pro[j+1];
                pro [j+1]= temp;
}}}
time= (time < at [comp]) ? at[comp] : time;
at [comp] = time +bt[comp];
```

```
tat[comp] = ct[comp] - at[comp];
wt[comp] = tat[comp] - bt[comp];
time = ct[comp];
comp++;
}
Printf("Process ArrivalTime BurstTime TurnAroundTime WaitingTime
        Priority");
for (int i=0; i<n; i++){
    Printf("%d %d %d %d %d %d\n", Proc[i], at[i],
            bt[i], tat[i], wt[i], priority[i]);

    total_tat += tat[i];
    total_wt += wt[i];
}

Printf("Average TurnAround Time : %.2f", total_tat/n);
Printf("Average Waiting Time : %.2f", total_wt/n);
return 0;
}
```

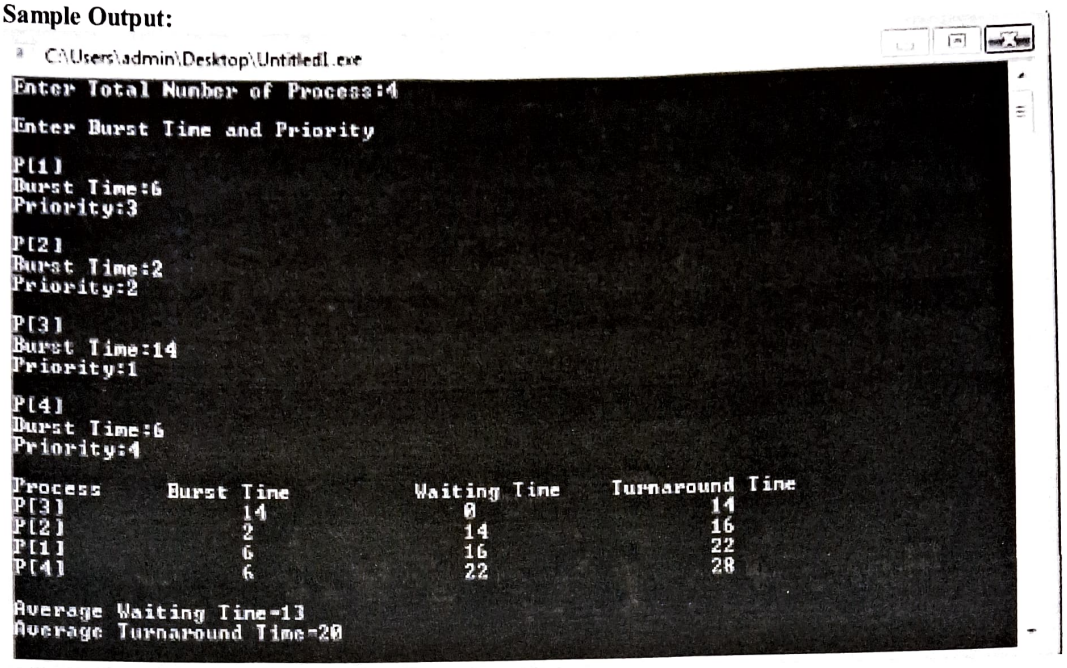
Gantt chart:



Tabulation:

Process	BT (ms)	Priority (ms)	AT (ms)	CT (ms)	TAT = CT-AT (ms)	WT=TAT-BT (ms)
1	7	2	0	17	17	10
2	3	3	0	20	20	17
3	10	1	0	10	10	0

Sample Output:



Enter the no. of processes: 3
Enter the process1 Burst time: 7
Enter the process2 Burst time: 3
Enter the process 3 Burst time: 10
Enter the priority of process1: 2
Enter the priority of process2: 3
Enter the priority of process 3: 1

Process	Burst time	Priority	Turn Around Time	Waiting Time
3	10	1	10	0
1	7	2	17	10
2	3	3	20	17

Average Turn Around Time: 15.66
Average Waiting Time: 9.00

Result:

Thus the implementation of priority cpu scheduling has been successfully executed.