Ex. No.: 6c **Date:** ***19.3.2025***

NAME:Prem Roshan P **ROLLNO:231901036**

PRIORITY SCHEDULING

Aim:

To implement the Priority Scheduling technique in C.

Algorithm:

1. Start the program.

2. Get the number of processes from the user.

3. Read the process name (or ID), burst time, and priority of each process. 4.

Sort the processes based on their priority (lower number = higher priority). 5.

Set the waiting time of the first process to 0.

6. For each remaining process: waiting_time[i] = waiting_time[i-1] + burst_time[i-1]

7. Calculate turnaround time: turnaround_time[i] = waiting_time[i] + burst_time[i] 8.

Compute the total and average waiting time and turnaround time.

9. Display the details.

10. End the program.

Program Code (in C):

```
#include <stdio.h>


int main() { int bt[20], p[20], wt[20],
tat[20], prio[20];
 int i, j, n, temp; float
total_wt = 0, total_tat = 0;


 printf("Enter the number of processes:\n");
scanf("%d", &n);
```

```
 printf("Enter Burst Time and Priority of each process:\n");
for (i = 0; i < n; i++) {
 printf("Process %d - Burst Time: ", i + 1); scanf("%d", &bt[i]);
printf("Process %d - Priority (lower number = higher priority): ", i + 1);
scanf("%d", &prio[i]);
 p[i] = i + 1;
 }

 // Sort processes based on priority
 for (i = 0; i < n - 1; i++) {
for (j = i + 1; j < n; j++) {
if (prio[i] > prio[j]) {
// Swap priority
temp = prio[i];
prio[i] = prio[j];
prio[j] = temp;

 // Swap burst
time temp = bt[i];
bt[i] = bt[j]; bt[j]
= temp;

 // Swap process
ID temp = p[i];
p[i] = p[j]; p[j] =
temp;
 }
 }
 }
 wt[0] = 0; for (i = 1; i <
n; i++) { wt[i] = wt[i -
 1] + bt[i - 1]; total_wt
+= wt[i];
 }

 for (i = 0; i < n; i++) {
tat[i] = wt[i] + bt[i];
total_tat += tat[i];
 }

 printf("\nProcess\tBurst Time\tPriority\tWaiting Time\tTurnaround Time\n");
for (i = 0; i < n; i++) {
 printf("%d\t%d\t\t%d\t\t%d\t\t%d\n", p[i], bt[i], prio[i], wt[i], tat[i]);   }

 printf("\nAverage Waiting Time: %.2f", total_wt / n);
printf("\nAverage Turnaround Time: %.2f\n", total_tat / n);

 return 0;
}
```

Sample Output:

```
Enter the number of processes:
4
Enter Burst Time and Priority of each process:
Process 1 - Burst Time: 10
Process 1 - Priority: 3
Process 2 - Burst Time: 1
Process 2 - Priority: 1
Process 3 - Burst Time: 2
Process 3 - Priority: 4

Process 4 - Burst Time: 1

Process 4 - Priority: 2


Process Burst Time Priority Waiting Time Turnaround Time  2 1 1 0 1

4 1 2 1 2

 1 10 3 2 12

3 2 4 12 14


Average Waiting Time: 3.75

Average Turnaround Time: 7.25
```

Result:

The Priority Scheduling algorithm was successfully implemented and tested. The program displayed correct waiting and turnaround times based on priority.