Program 4: Priority Scheduling

```
// C++ program for implementation of FCFS
// scheduling
#include <bits/stdc++.h>
using namespace std;

struct Process {
int pid; // Process ID
int bt; // CPU Burst time required
int priority; // Priority of this process
};

// Function to sort the Process acc. to priority
bool comparison(Process a, Process b)
{
return (a.priority > b.priority);
}

// Function to find the waiting time for all
// processes
void findWaitingTime(Process proc[], int n, int wt[])
{
// waiting time for first process is 0
wt[0] = 0;

// calculating waiting time
```

```
for (int i = 1; i < n; i++)

wt[i] = proc[i - 1].bt + wt[i - 1];


}


// Function to calculate turn around time

void findTurnAroundTime(Process proc[], int n, int wt[],

int tat[])


{

// calculating turnaround time by adding

// bt[i] + wt[i]

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

tat[i] = proc[i].bt + wt[i];


}


// Function to calculate average time

void findavgTime(Process proc[], int n)

{

int wt[n], tat[n], total_wt = 0, total_tat = 0;


// Function to find waiting time of all processes

findWaitingTime(proc, n, wt);


// Function to find turn around time for all processes

findTurnAroundTime(proc, n, wt, tat);
```

```
// Display processes along with all details
cout << "\nProcesses "
<< " Burst time "
<< " Waiting time "
<< " Turn around time\n";

// Calculate total waiting time and total turn
// around time
for (int i = 0; i < n; i++) {
total_wt = total_wt + wt[i];

total_tat = total_tat + tat[i];
cout << " " << proc[i].pid << "\t\t" << proc[i].bt
<< "\t " << wt[i] << "\t\t " << tat[i]
<< endl;

}

cout << "\nAverage waiting time = "
<< (float)total_wt / (float)n;
cout << "\nAverage turn around time = "
<< (float)total_tat / (float)n;

}

void priorityScheduling(Process proc[], int n)
```

```
{
// Sort processes by priority
sort(proc, proc + n, comparison);

cout << "Order in which processes gets executed \n";
for (int i = 0; i < n; i++)
cout << proc[i].pid << " ";

findavgTime(proc, n);
}

// Driver code
int main()
{
Process proc[]
= { { 1, 10, 2 }, { 2, 5, 0 }, { 3, 8, 1 } };
int n = sizeof proc / sizeof proc[0];
priorityScheduling(proc, n);


return 0;
}
```

OUTPUT:

```
Order in which processes gets executed
1 3 2
```

Processes Burst time Waiting time Turn around time

1 10 0 10

3 8 10 18

2 5 18 23

Average waiting time = 9.33333

Average turn around time = 17