**Experiment No 5**

**Aim -: To write a program to implement CPU scheduling algorithm to perform prority scheduling**

#include<stdio.h>

Struct process {

Int pid;

Int arrival\_time;

Int burst\_time;

Int priority;

Int waiting\_time;

Int turnaround\_time;

};

Void main() {

Int n, I, j;

Struct process proc[10], temp;

Float total\_waiting\_time = 0, total\_turnaround\_time = 0;

Printf(“Enter the number of processes: “);

Scanf(“%d”, &n);

// Input process details

For(i=0; i<n; i++) {

Printf(“\nEnter the process id, arrival time, burst time and priority for process %d: “, i+1);

Scanf(“%d%d%d%d”, &proc[i].pid, &proc[i].arrival\_time, &proc[i].burst\_time, &proc[i].priority);

}

// Sort processes by priority

For(i=0; i<n; i++) {

For(j=i+1; j<n; j++) {

If(proc[i].priority < proc[j].priority) {

Temp = proc[i];

Proc[i] = proc[j];

Proc[j] = temp;

}

}

}

// Calculate waiting and turnaround time for each process

Proc[0].waiting\_time = 0;

Proc[0].turnaround\_time = proc[0].burst\_time;

For(i=1; i<n; i++) {

Proc[i].waiting\_time = proc[i-1].waiting\_time + proc[i-1].burst\_time;

Proc[i].turnaround\_time = proc[i].waiting\_time + proc[i].burst\_time;

}

// Calculate total waiting and turnaround time

For(i=0; i<n; i++) {

Total\_waiting\_time += proc[i].waiting\_time;

Total\_turnaround\_time += proc[i].turnaround\_time;

}

// Print process details and scheduling metrics

Printf(“\nProcess\tArrival Time\tBurst Time\tPriority\tWaiting Time\tTurnaround Time”);

For(i=0; i<n; i++) {

Printf(“\n%d\t\t%d\t\t%d\t\t%d\t\t%d\t\t%d”, proc[i].pid, proc[i].arrival\_time, proc[i].burst\_time, proc[i].priority, proc[i].waiting\_time, proc[i].turnaround\_time);

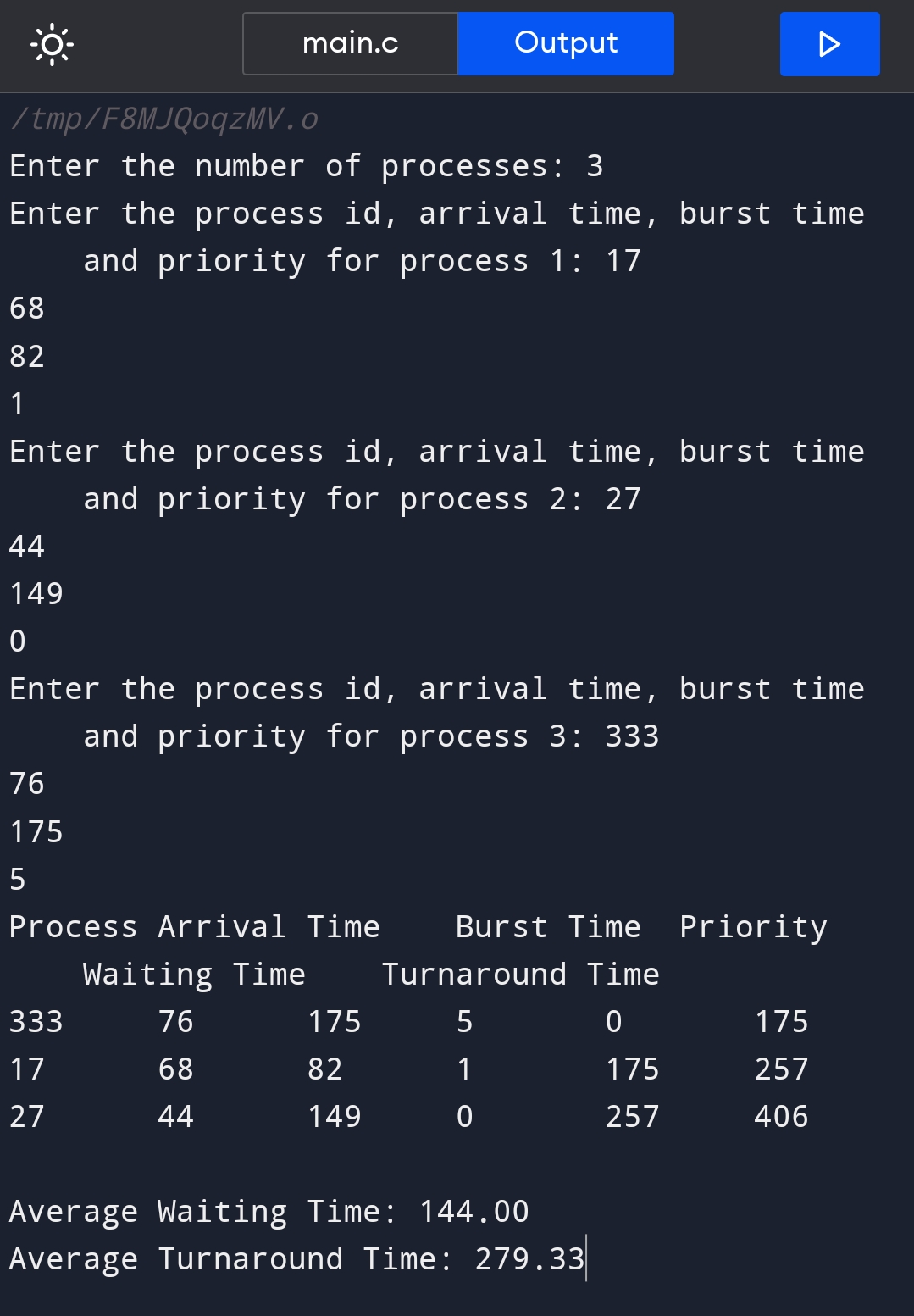
}

Printf(“\n\nAverage Waiting Time: %.2f”, total\_waiting\_time/n);

Printf(“\nAverage Turnaround Time: %.2f”, total\_turnaround\_time/n);

}

**Output -:**

****