**Lab Report #07**

**Experiment Name #**  Short Job First (SJF) preemptive Scheduling Algorithm.

**Aim and Objective:**

The purpose of the experiment is to learn Short Job First (SJF) preemptive scheduling algorithm and execute a code by using C.

* To learn SJF preemptive scheduling algorithm.
* To implement SJF using C.

**Code:**

#include <stdio.h>

int main()

{

int arrival\_time[10], burst\_time[10], temp[10];

int i, smallest, count = 0, time, limit;

double wait\_time = 0, turnaround\_time = 0, end;

float average\_waiting\_time, average\_turnaround\_time;

printf("\nTotal Number of Processes:\t");

scanf("%d", &limit);

for(i = 0; i < limit; i++)

{

printf("\nArrival Time:\t");

scanf("%d", &arrival\_time[i]);

printf("Burst Time:\t");

scanf("%d", &burst\_time[i]);

temp[i] = burst\_time[i];

}

burst\_time[9] = 9999;

for(time = 0; count != limit; time++)

{

smallest = 9;

for(i = 0; i < limit; i++)

{

if(arrival\_time[i] <= time && burst\_time[i] < burst\_time[smallest] && burst\_time[i] > 0)

{

smallest = i;

}

}

burst\_time[smallest]--;

if(burst\_time[smallest] == 0)

{

count++;

end = time + 1;

wait\_time = wait\_time + end - arrival\_time[smallest] - temp[smallest];

turnaround\_time = turnaround\_time + end - arrival\_time[smallest];

}

}

average\_waiting\_time = wait\_time / limit;

average\_turnaround\_time = turnaround\_time / limit;

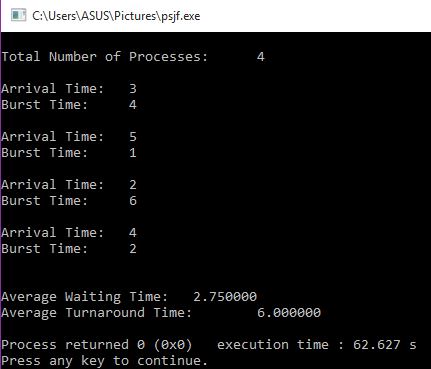
printf("\n\nAverage Waiting Time:\t%lf\n", average\_waiting\_time);

printf("Average Turnaround Time:\t%lf\n", average\_turnaround\_time);

return 0;

}

**Output:**



**Conclusion:**

By following these steps one can perform short job first preemptive using c. In this algorithm the process which has shortest burst time has to do the job first and does not hold CPU and CPU can go to another process if it arrives in the execution and has shortest burst time. It is quite important CPU scheduling in operation system.