SHORTEST JOB FIRST

Aim:

To implement the Shortest Job First(SJF) scheduling technique

Algorithm:

1. Declare the structure and its elements.
2. Get number of processes as input from the user.
3. Read the process name, arrival time and burst time
4. Initialize waiting time, turnaround time & flag of read processes to zero.
5. Sort based on burst time of all processes in ascending order
6. Calculate the waiting time and turnaround time for each process.
7. Calculate the average waiting time and average turnaround time.
8. Display the results.

Program Code:

```
#include<studio.h>
int main()
{
	int n, i, j;
	float avg_waiting_time = 0, avg_turnaround_time = 0;
	printf("Enter the number of processes: ");
	scanf("%d", &n);
	struct Process processes[n];
	for (i = 0; i< n; i++) {
	printf("Enter arrival time and burst time of process %d: ", i+1);
	scanf("%d %d", &processes[i].arrival_time, &processes[i].burst_time);
}
	qsort(processes, n, sizeof(struct Process), compare);
	processes[0].waiting_time = 0;
	for (i = 1; i< n; i++)
	{
		processes[i].waiting_time = 0;
		for (j = 0; j <i; j++)
		{
			processes[i].waiting_time += processes[j].burst_time;
		}
		avg_waiting_time += processes[i].waiting_time;
	}
	avg_waiting_time /= n;
```

```
        for (i = 0; i< n; i++)
        {

        avg_turnaround_time += processes[i].burst_time + processes[i].waiting_ti
me;
        }
        avg_turnaround_time /= n;
        printf("\nProcess\tArrival Time\tBurst Time\tWaiting Time\tTurnaround Ti
me\n");
        for (i = 0; i< n; i)
        {

        printf("%d\t\t%d\t\t%d\t\t%d\t\t%d\n", i+1, processes[i].arrival_time, proce
sses[i].burst_time, processes[i].waiting_time, processes[i].burst_time+processes[i
].waiting_time);
        }
        printf("\nAverage Waiting Time: %f\n", avg_waiting_time);
        printf("Average Turnaround Time: %f\n", avg_turnaround_time);
        return 0;
}
```