Lab Exercise – 4

- AIM :: WAP in C to implement CPU scheduling for `shortest job first` (sjf).

Source_Code ::

```
#include <stdio.h>


typedef struct
{
  int pid;        // Process ID
  int arrival;    // Arrival time
  int burst;      // Burst time
  int completion; // Completion time
  int waiting;    // Waiting time
  int turnaround; // Turnaround time
} Process;


// Function to sort processes by arrival time, and by burst time in case of tie
void sortByArrival(Process *p, int n)
{
  for (int i = 0; i < n - 1; i++)
  {
    for (int j = 0; j < n - i - 1; j++)
    {
```

```
      if (p[j].arrival > p[j + 1].arrival ||
          (p[j].arrival == p[j + 1].arrival && p[j].burst > p[j + 1].burst))
      {
        Process temp = p[j];
        p[j] = p[j + 1];
        p[j + 1] = temp;
      }
    }
  }
}

// Main SJF logic
void sjfScheduling(Process *p, int n)
{
  int time = 0, completed = 0, minIndex;

  while (completed < n)
  {
    minIndex = -1;

    // Find process with min burst time from the pool of arrived processes
    for (int i = 0; i < n; i++)
    {
      if (p[i].arrival <= time && p[i].completion == 0)
      {
        if (minIndex == -1 || p[i].burst < p[minIndex].burst)
```

```
        {
          minIndex = i;
        }
      }
    }

    if (minIndex != -1)
    {
      if (time < p[minIndex].arrival)
        time = p[minIndex].arrival; // Set time to the process arrival time if idle

      time += p[minIndex].burst;
      p[minIndex].completion = time;
      p[minIndex].turnaround = p[minIndex].completion - p[minIndex].arrival;
      p[minIndex].waiting = p[minIndex].turnaround - p[minIndex].burst;
      completed++;
    }
    else
    {
      time++;
    }
  }
}

// Function to display the Gantt chart
void displayGanttChart(Process *p, int n)
```

```
{
  int startTime = p[0].arrival;
  printf("Gantt Chart:\n%d", startTime);

  for (int i = 0; i < n; i++)
  {
    printf(" -- P%d -- %d", p[i].pid, p[i].completion);
  }

  printf("\n\n");
}

// Function to calculate and display average times
void calculateAverages(Process *p, int n)
{
  float totalTurnaround = 0, totalWaiting = 0;

  for (int i = 0; i < n; i++)
  {
    totalTurnaround += p[i].turnaround;
    totalWaiting += p[i].waiting;
  }

  printf("\nAverage Turnaround Time: %.2f\n", totalTurnaround / n);
  printf("Average Waiting Time: %.2f\n", totalWaiting / n);
}
```

```
// Function to display process information
void displayResults(Process *p, int n)
{
  printf("PID\tArrival\t Burst\t Completion\tTurnaround\tWaiting\n");
  for (int i = 0; i < n; i++)
  {
    printf("%d\t%d\t %d\t %d\t\t%d\t\t%d\n", p[i].pid, p[i].arrival, p[i].burst,
          p[i].completion, p[i].turnaround, p[i].waiting);
  }
}

int main()
{
  int n;
  printf("\n5C6 - Amit Singhal (11614802722)\n");
  printf("\nEnter number of processes: ");
  scanf("%d", &n);
  Process p[n];

  for (int i = 0; i < n; i++) {
    printf("\nEnter Arrival Time and Burst Time for Process %d: ", i + 1);
    p[i].pid = i + 1;
    scanf("%d%d", &p[i].arrival, &p[i].burst);
    p[i].completion = 0; // Initially, no process is completed
  }
```

```
    printf("\n");

    sortByArrival(p, n);
    sjfScheduling(p, n);
    displayGanttChart(p, n);
    displayResults(p, n);
    calculateAverages(p, n);

    printf("\n");

    return 0;
}
```

Output ::

```
singhal-amit@singhal-amit-ThinkPad-T430:~/Downloads/_LAB_Work/OS/Code$ vi prg_4_sjf.c
singhal-amit@singhal-amit-ThinkPad-T430:~/Downloads/_LAB_Work/OS/Code$ gcc prg_4_sjf.c
singhal-amit@singhal-amit-ThinkPad-T430:~/Downloads/_LAB_Work/OS/Code$ ./a.out

5C6 - Amit Singhal (11614802722)

Enter number of processes: 4

Enter Arrival Time and Burst Time for Process 1: 1 3

Enter Arrival Time and Burst Time for Process 2: 2 4

Enter Arrival Time and Burst Time for Process 3: 1 2

Enter Arrival Time and Burst Time for Process 4: 4 4

Gantt Chart:
1 -- P3 -- 3 -- P1 -- 6 -- P2 -- 10 -- P4 -- 14

PID     Arrival  Burst   Completion     Turnaround      Waiting
3       1        2       3              2               0
1       1        3       6              5               2
2       2        4       10             8               4
4       4        4       14             10              6

Average Turnaround Time: 6.25
Average Waiting Time: 3.00
```