Lab Exercise – 3

- AIM :: WAP in C to implement CPU scheduling for first come first serve.

Source_Code ::

```
#include <stdio.h>
#include <stdlib.h>
#define MAX 100

typedef struct
{
  int pid;
  int burst_time;
  int waiting_time;
  int turnaround_time;
} Process;

void print_table(Process p[], int n);
void print_gantt_chart(Process p[], int n);

int main()
{
  Process p[MAX];
  int i, j, n;
  int sum_waiting_time = 0, sum_turnaround_time = 0;
```

```
printf("\n5C6 - Amit Singhal (11614802722) \n");

printf("\nEnter total number of processes: ");
scanf("%d", &n);

printf("\nEnter burst time for each process:\n");
for (i = 0; i < n; i++)
{
  p[i].pid = i + 1;
  printf("P[%d]: ", i + 1);
  scanf("%d", &p[i].burst_time);
  p[i].waiting_time = 0;
  p[i].turnaround_time = 0;
}

p[0].turnaround_time = p[0].burst_time;
for (i = 1; i < n; i++)
{
  p[i].waiting_time = p[i - 1].waiting_time + p[i - 1].burst_time;
  p[i].turnaround_time = p[i].waiting_time + p[i].burst_time;
}

for (i = 0; i < n; i++)
{
  sum_waiting_time += p[i].waiting_time;
  sum_turnaround_time += p[i].turnaround_time;
```

```
  }


  puts("");

  print_table(p, n);

  printf("\nTotal Waiting Time: %d\n", sum_waiting_time);

  printf("Average Waiting Time: %.2lf\n", (double)sum_waiting_time / (double)n);

  printf("Total Turnaround Time: %d\n", sum_turnaround_time);

  printf("Average Turnaround Time: %.2lf\n", (double)sum_turnaround_time /
(double)n);


  puts("\nGANTT CHART\n");

  print_gantt_chart(p, n);


  return 0;

}


void print_table(Process p[], int n)

{

  int i;


  puts("+-----+------------+--------------+-----------------+");

  puts("| PID | Burst Time | Waiting Time | Turnaround Time |");

  puts("+-----+------------+--------------+-----------------+");


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

  {

    printf("| %3d | %10d | %12d | %15d |\n", p[i].pid, p[i].burst_time, p[i].waiting_time,
p[i].turnaround_time);
```

```
    puts("+-----+------------+-------------+----------------+");
  }
}

void print_gantt_chart(Process p[], int n)
{
  int i, j;

  // Top border of the Gantt chart
  puts("+----+----+----+----+----+----+");

  // Process IDs
  puts("| P1 | P2 | P3 | P4 | P5 | P6 |");

  // Bottom border of the Gantt chart
  puts("+----+----+----+----+----+----+");

  for (i = 0; i < n; i++)
  {
    printf("| %d  ", p[i].turnaround_time);
    if (p[i].turnaround_time > 9)
      printf("\b"); // Remove 1 space if the number has 2 digits
  }
  printf("|\n");
  puts("+----+----+----+----+----+----+");
  printf("\n");
}
```

Output ::

```
singhal-amit@singhal-amit-ThinkPad-T430:~/Downloads/_LAB_Work/OS/Code$ vi prg_3_fcfs.c
singhal-amit@singhal-amit-ThinkPad-T430:~/Downloads/_LAB_Work/OS/Code$ gcc prg_3_fcfs.c -o a
singhal-amit@singhal-amit-ThinkPad-T430:~/Downloads/_LAB_Work/OS/Code$ ./a

SC6 - Amit Singhal (11614802722)

Enter total number of processes: 6

Enter burst time for each process:
P[1]: 5
P[2]: 3
P[3]: 9
P[4]: 8
P[5]: 4
P[6]: 7

+-----+------------+--------------+-----------------+
| PID | Burst Time | Waiting Time | Turnaround Time |
+-----+------------+--------------+-----------------+
|   1 |          5 |            0 |               5 |
+-----+------------+--------------+-----------------+
|   2 |          3 |            5 |               8 |
+-----+------------+--------------+-----------------+
|   3 |          9 |            8 |              17 |
+-----+------------+--------------+-----------------+
|   4 |          8 |           17 |              25 |
+-----+------------+--------------+-----------------+
|   5 |          4 |           25 |              29 |
+-----+------------+--------------+-----------------+
|   6 |          7 |           29 |              36 |
+-----+------------+--------------+-----------------+

Total Waiting Time: 84
Average Waiting Time: 14.00
Total Turnaround Time: 120
Average Turnaround Time: 20.00

GANTT CHART

+----+----+----+----+----+----+
| P1 | P2 | P3 | P4 | P5 | P6 |
+----+----+----+----+----+----+
| 5  | 8  | 17 | 25 | 29 | 36 |
+----+----+----+----+----+----+
```