Lab Exercise – 3

- AIM :: WAP in C to implement CPU scheduling for `first come first serve` (fcfs).

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
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)
```

```
      {
        Process temp = p[j];
        p[j] = p[j + 1];
        p[j + 1] = temp;
      }
    }
  }
}

// Main FCFS logic
void fcfsScheduling(Process *p, int n)
{
  int time = 0;

  for (int i = 0; i < n; i++)
  {
    if (time < p[i].arrival)
      time = p[i].arrival; // Set time to the process arrival time if idle

    time += p[i].burst;
    p[i].completion = time;
    p[i].turnaround = p[i].completion - p[i].arrival;
    p[i].waiting = p[i].turnaround - p[i].burst;
  }
}

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

```
  int currentTime = p[0].arrival; // Start from the first process arrival time
  printf("Gantt Chart:\n");

  // Print initial time
  printf("%d", currentTime);

  for (int i = 0; i < n; i++)
  {
    if (currentTime < p[i].arrival)
    {
      // Display idle time
      printf(" -- XX -- %d", p[i].arrival);
      currentTime = p[i].arrival; // Update current time to the arrival of the next process
    }
    // Display the process and its completion time
    printf(" -- P%d -- %d", p[i].pid, p[i].completion);
    currentTime = p[i].completion; // Update current time to the completion of the current process
  }

  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);
  fcfsScheduling(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_3_fcfs.c
singhal-amit@singhal-amit-ThinkPad-T430:~/Downloads/_LAB_Work/OS/Code$ gcc prg_3_fcfs.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: 0 2

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

Enter Arrival Time and Burst Time for Process 3: 5 3

Enter Arrival Time and Burst Time for Process 4: 6 4

Gantt Chart:
0 -- P1 -- 2 -- P2 -- 4 -- XX -- 5 -- P3 -- 8 -- P4 -- 12

PID     Arrival  Burst   Completion     Turnaround      Waiting
1       0        2       2              2               0
2       1        2       4              3               1
3       5        3       8              3               0
4       6        4       12             6               2

Average Turnaround Time: 3.50
Average Waiting Time: 0.75
```