Program for FCFS CPU Scheduling

- **Arrival Time:** The time at which the process arrives in the ready queue.
- **Completion Time:** The time at which the process completes its execution.
- **Turn Around Time:** Time Difference between completion time and arrival time. Turn Around Time = (Completion Time – Arrival Time)
- **Waiting Time (W. T):** Time Difference between turnaround time and burst time

  Waiting Time = (Turn Around Time – Burst Time).

Implementation

```
1 - Input the processes along with their burst time (bt).
2 - Find waiting time (wt) for all processes.
3 - As first process that comes need not to wait so
    waiting time for process 1 will be 0 i.e. wt[0] = 0.
4 - Find waiting time for all other processes i.e. for
     process i ->
      wt[i] = bt[i-1] + wt[i-1] .
5 - Find turnaround time = waiting_time + burst_time
    for all processes.
6 - Find average waiting time =
                 total_waiting_time / no_of_processes.
7 - Similarly, find average turnaround time =
                 total_turn_around_time / no_of_processes.
```

Ex 1: Code for FCFS operation

$ touch fcfs2.c

```
#include <stdio.h>

// Function to compute the waiting time for each process
void findWaitingTime(int processes[], int n, int burst_time[], int waiting_time[]) {
        waiting_time[0] = 0;  // The waiting time for the first process is always 0

        // Compute waiting time for each subsequent process
        for (int i = 1; i < n; i++) {
        waiting_time[i] = burst_time[i - 1] + waiting_time[i - 1];
        }
}
```

```
// Function to compute the turnaround time for each process
void findTurnAroundTime(int processes[], int n, int burst_time[], int waiting_time[], int
turnaround_time[]) {
        // Turnaround time is the sum of burst time and waiting time for each process
        for (int i = 0; i < n; i++) {
        turnaround_time[i] = burst_time[i] + waiting_time[i];
        }
}

// Function to compute and display average waiting and turnaround times
void findavgTime(int processes[], int n, int burst_time[]) {
        int waiting_time[n], turnaround_time[n];
        int total_waiting_time = 0, total_turnaround_time = 0;

        // Calculate waiting time and turnaround time for all processes
        findWaitingTime(processes, n, burst_time, waiting_time);
        findTurnAroundTime(processes, n, burst_time, waiting_time, turnaround_time);

        // Display process details
        printf("Processes   Burst time   Waiting time   Turnaround time\n");

        // Calculate total waiting time and total turnaround time
        for (int i = 0; i < n; i++) {
        total_waiting_time += waiting_time[i];
        total_turnaround_time += turnaround_time[i];
        printf("   %d              %d                %d                %d\n", processes[i], burst_time[i],
waiting_time[i], turnaround_time[i]);
        }

        // Compute and display average waiting time and turnaround time
        float avg_waiting_time = (float)total_waiting_time / n;
        float avg_turnaround_time = (float)total_turnaround_time / n;
        printf("Average waiting time = %.2f\n", avg_waiting_time);
        printf("Average turnaround time = %.2f\n", avg_turnaround_time);
}


int main() {
        int n;

        // Prompt the user to enter the number of processes

        printf("Enter the number of processes: ");
        scanf("%d", &n);

        int processes[n];  // Array to store process IDs
        int burst_time[n]; // Array to store burst times for each process
```

```
    // Get the burst time for each process from the user
    for (int i = 0; i < n; i++) {
    processes[i] = i + 1;  // Assign process IDs starting from 1
    printf("Enter burst time for process %d: ", processes[i]);
    scanf("%d", &burst_time[i]);
    }

    // Calculate and display average waiting time and turnaround time
    findavgTime(processes, n, burst_time);

    return 0;
}
```

$ gcc fcfs2.c -o fcfs2

$ ./fcfs2

Output:

```
Enter the number of processes: 4
Enter burst time for process 1: 11
Enter burst time for process 2: 21
Enter burst time for process 3: 33
Enter burst time for process 4: 41
Processes   Burst time   Waiting time   Turnaround time
  1      11      0                 11
  2      21                11                32
  3      33                32                65
  4      41                65                106
Average waiting time = 27.00
Average turnaround time = 53.50
```

Ex 2. Fcfs scheduling code 2

$ touch fcfs4.c

```
#include <stdio.h>

int main() {
    int p[10], at[10], bt[10], ct[10], tat[10], wt[10];
    int i, j, temp = 0, n;
    float awt = 0, atat = 0;
}

// Get the number of processes
printf("Enter number of processes: ");
scanf("%d", &n);

// Get the process IDs
printf("Enter %d process IDs: ", n);
for (i = 0; i < n; i++) {
scanf("%d", &p[i]);
}

// Get the arrival times
printf("Enter %d arrival times: ", n);
for (i = 0; i < n; i++) {
scanf("%d", &at[i]);
}

// Get the burst times
printf("Enter %d burst times: ", n);
for (i = 0; i < n; i++) {
scanf("%d", &bt[i]);
}

// Sort processes based on arrival times (using Bubble Sort)
for (i = 0; i < n - 1; i++) {
for (j = 0; j < n - i - 1; j++) {
if (at[j] > at[j + 1]) {
        // Swap process IDs
        temp = p[j];
        p[j] = p[j + 1];
        p[j + 1] = temp;

        // Swap arrival times
        temp = at[j];
        at[j] = at[j + 1];
        at[j + 1] = temp;

        // Swap burst times
        temp = bt[j];
        bt[j] = bt[j + 1];
        bt[j + 1] = temp;
}
}
}

// Calculate completion time for the first process
ct[0] = at[0] + bt[0];
```

```
        // Calculate completion times for the rest of the processes
        for (i = 1; i < n; i++) {
        // If the CPU is idle until the next process arrives
        int idle_time = 0;
        if (ct[i - 1] < at[i]) {
        idle_time = at[i] - ct[i - 1];
        }
        ct[i] = ct[i - 1] + bt[i] + idle_time;
        }

        // Calculate turnaround time and waiting time
        printf("\nProcess\tA.T\tB.T\tC.T\tTAT\tWT");
        for (i = 0; i < n; i++) {
        tat[i] = ct[i] - at[i];
        wt[i] = tat[i] - bt[i];
        atat += tat[i];
        awt += wt[i];
        printf("\nP%d\t%d\t%d\t%d\t%d\t%d", p[i], at[i], bt[i], ct[i], tat[i], wt[i]);
        }

        // Calculate and display average turnaround time and waiting time
        atat /= n;
        awt /= n;
        printf("\nAverage turnaround time = %.2f", atat);
        printf("\nAverage waiting time = %.2f", awt);

        return 0;
}
```

Output:

```
$ gcc fcfs4.c -o fcfs4

$ ./fcfs4

enter no of proccess you want:3
enter 3 process:33
2
11
enter 3 arrival time:3
1
2
enter 3 burst time:2
4
6
```

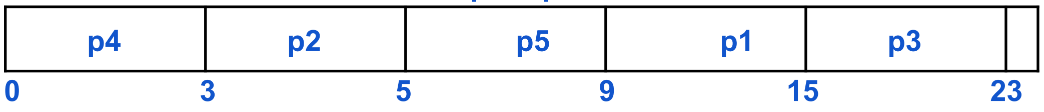
```
p       A.T     B.T     C.T     TAT     WT
P2      1       4       5       4       0
P0      1       0       5       4       4
P11     2       6       11      9       3

average turnaround time is 5.666667

average wating timme is 2.333333
```

SJF scheduling :

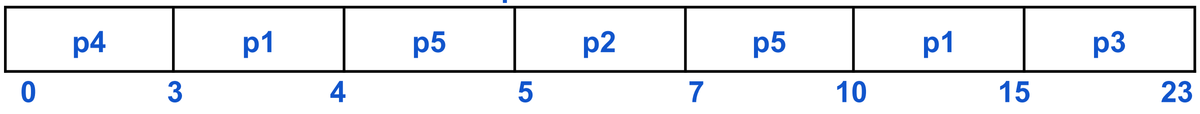
process	Burst Time	Arrival Time4
p1	6	2
p2	2	5
p3	8	1
p4	3	0
p5	4	4

Gannt chart

Non-premptive SJF



Preemptive SJF


p4
p1
p5
p2
p5
p1
p3
0
3
4
5
7
10
15
23