# OS LAB 1

**AIM:** Write a C program to simulate the following non pre-emptive CPU scheduling algorithm to find turnaround time and waiting time.

1. FCFS
2. SJF (pre-emptive & Non pre-emptive)

## SOURCE CODE

```
#include<stdio.h>
#include<stdlib.h>
void findWaitingTime(int processes[], int n, int burstTime[], int waitingTime[])
{
   waitingTime[0] = 0;

   for (int i = 1; i < n ; i++ )
      waitingTime[i] =  burstTime[i-1] + waitingTime[i-1] ;
}

void findTurnaroundTime( int processes[], int n, int burstTime[], int waitingTime[], int
turnaroundTime[])
{
   for (int i = 0; i < n ; i++)
      turnaroundTime[i] = burstTime[i] + waitingTime[i];
}

void findAverageTime( int processes[], int n, int burstTime[])
{
   int waitingTime[n], turnaroundTime[n], totalWaitingTime = 0, totalTurnaroundTime = 0;

   findWaitingTime(processes, n, burstTime, waitingTime);

   findTurnaroundTime(processes, n, burstTime, waitingTime, turnaroundTime);
```

```
    printf("\nProcess\tBurst Time\tWaiting Time\tTurnaround Time\n");

    for (int i=0; i<n; i++)
    {
        totalWaitingTime = totalWaitingTime + waitingTime[i];
        totalTurnaroundTime = totalTurnaroundTime + turnaroundTime[i];
        printf("%d\t%d\t\t%d\t\t%d\n", i+1, burstTime[i], waitingTime[i], turnaroundTime[i]);
    }

    printf("\nAverage Waiting Time: %.2f", (float)totalWaitingTime/n);
    printf("\nAverage Turnaround Time: %.2f", (float)totalTurnaroundTime/n);
}

void fcfsScheduling(int processes[], int n, int burstTime[])
{
    printf("\nFirst-Come, First-Served (FCFS) Scheduling:\n");
    findAverageTime(processes, n, burstTime);
}

void sjfNonPreemptiveScheduling(int processes[], int n, int burstTime[])
{
    // Sort the burst time in ascending order
    for (int i = 0; i < n-1; i++)
    {
        for (int j = 0; j < n-i-1; j++)
        {
            if (burstTime[j] > burstTime[j+1])
            {
                int temp = burstTime[j];
                burstTime[j] = burstTime[j+1];
                burstTime[j+1] = temp;
```

```
            temp = processes[j];
            processes[j] = processes[j+1];
            processes[j+1] = temp;
        }
      }
   }

   printf("\nShortest Job First (SJF) Non-Preemptive Scheduling:\n");
   findAverageTime(processes, n, burstTime);
}

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

   int processes[n], burstTime[n];

   for (int i = 0; i < n; i++)
   {
      printf("Enter burst time for process %d: ", i+1);
      scanf("%d", &burstTime[i]);
      processes[i] = i+1;
   }

   printf("\nChoose the scheduling algorithm:\n");
   printf("1. First-Come-First-Served (FCFS)\n");
   printf("2. Shortest Job First (SJF) Non-Preemptive\n");
   printf("Enter your choice: ");
   scanf("%d", &choice);
```

```
    switch(choice)
    {
        case 1:
            fcfsScheduling(processes, n, burstTime);
            break;

        case 2:
            sjfNonPreemptiveScheduling(processes, n, burstTime);
            break;

        default:
            printf("Invalid choice. Exiting...\n");
            return 0;
    }

    return 0;
}
```

# OUTPUT SCREENSHOTS