

3. Design a CPU scheduling program with C using First Come First Served technique with the following considerations.

a. All processes are activated at time 0.

b. Assume that no process waits on I/O devices.

```
#include <stdio.h>
```

```
int main()
```

```
    int n, bt[20], wt[20], tat[20];
```

```
    int i, total_wt = 0, total_tat = 0;
```

```
    printf("Enter number of processes: ");
```

```
    scanf("%d", &n);
```

```
    printf("Enter burst time of each process:\n");
```

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

```
        printf("P%d: ", i + 1);
```

```
        scanf("%d", &bt[i]);
```

```
    }
```

```
    wt[0] = 0;
```

```
    for (i = 1; i < n; i++)
```

```
        wt[i] = wt[i - 1] + bt[i - 1];
```

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

```
        tat[i] = wt[i] + bt[i];
```

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

```
        total_wt += wt[i];
```

```
        total_tat += tat[i];
```

```
    }
```

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

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

```
        printf("P%d\t%d\t%d\t%d\n", i + 1, bt[i], wt[i], tat[i]);
```

```
    printf("\nAverage Waiting Time = %.2f", (float)total_wt / n);
```

```
    printf("\nAverage Turnaround Time = %.2f\n", (float)total_tat / n);
```

```
    return 0;
```

```
}
```

Output

```
1 #include <stdio.h>
2 int main() {
3     int n, bt[20], wt[20], tat[20];
4     int i, total_wt = 0, total_tat = 0;
5     printf("Enter number of processes: ");
6     scanf("%d", &n);
7     printf("Enter burst time of each process:\n");
8     for (i = 0; i < n; i++) {
9         printf("P%d: ", i + 1);
10        scanf("%d", &bt[i]);
11    }
12    wt[0] = 0;
13    for (i = 1; i < n; i++)
14        wt[i] = wt[i - 1] + bt[i - 1];
15    for (i = 0; i < n; i++)
16        tat[i] = wt[i] + bt[i];
17    for (i = 0; i < n; i++) {
18        total_wt += wt[i];
19        total_tat += tat[i];
20    }
21    printf("\nProcess\tBurst\tWaiting\tTurnaround\n");
22    for (i = 0; i < n; i++)
23        printf("P%d\t%d\t%d\t%d\n", i + 1, bt[i], wt[i], tat[i]);
24 }
```

Enter number of processes: 3
Enter burst time of each process:
P1: 5
P2: 3
P3: 2

Process	Burst	Waiting	Turnaround
P1	5	0	5
P2	3	5	8
P3	2	8	10

Average Waiting Time = 4.33
Average Turnaround Time = 7.67

Process exited after 18.42 seconds with return value 0
Press any key to continue . . .

