Programiz C Online Compiler

main.c

Share Run

```c
#include <stdio.h>
struct Process {
    int id;
    int burst_time;
    int waiting_time;
    int turnaround_time;
};
void findWaitingTime(struct Process proc[], int n) {
    proc[0].waiting_time = 0;

    for (int i = 1; i < n; i++) {
        proc[i].waiting_time = proc[i - 1].waiting_time + proc[i - 1].burst_time;
    }
}
void findTurnaroundTime(struct Process proc[], int n) {
    for (int i = 0; i < n; i++) {
        proc[i].turnaround_time = proc[i].burst_time + proc[i].waiting_time;
    }
}
void findavgTime(struct Process proc[], int n) {
    findWaitingTime(proc, n);
    findTurnaroundTime(proc, n);
    float total_waiting_time = 0, total_turnaround_time = 0;
    for (int i = 0; i < n; i++) {
        total_waiting_time += proc[i].waiting_time;
        total_turnaround_time += proc[i].turnaround_time;
    }
```

Output

```
Average waiting time: 7.00
Average turnaround time: 13.00


=== Code Execution Successful ===
```

main.c

Share Run

```c
#include <stdio.h>
void findWaitingTime(int processes[], int n, int bt[], int wt[], int quantum) {
    int rem_bt[n];
    for (int i = 0; i < n; i++)
        rem_bt[i] = bt[i];
    int t = 0;
    while (1) {
        int done = 1;
        for (int i = 0; i < n; i++) {
            if (rem_bt[i] > 0) {
                done = 0;
                if (rem_bt[i] > quantum) {
                    t += quantum;
                    rem_bt[i] -= quantum;
                } else {
                    t = t + rem_bt[i];
                    wt[i] = t - bt[i];
                    rem_bt[i] = 0;
                }
            }
        }
        if (done == 1)
            break;
    }
}
void findTurnAroundTime(int processes[], int n, int bt[], int wt[], int tat[]) {
    }
```

Output

```
Average waiting time: 19.25
Average turnaround time: 28.00


=== Code Execution Successful ===
```