main.c

Share | Run

```c
#include <stdio.h>
#include <stdlib.h>
typedef struct {
    int pid;
    int bt;
    int wt;
    int tat;
} Process;
int compare(const void *a, const void *b) {
    return ((Process *)a)->bt - ((Process *)b)->bt;}
int main() {
    int n;
    printf("Enter number of processes: ");
    scanf("%d", &n);
    Process proc[n];
    for (int i = 0; i < n; i++) {
        proc[i].pid = i + 1;
        printf("Enter burst time for process %d: ", proc[i].pid);
        scanf("%d", &proc[i].bt);}
    qsort(proc, n, sizeof(Process), compare);
    proc[0].wt = 0;
    for (int i = 1; i < n; i++) {
        proc[i].wt = proc[i - 1].wt + proc[i - 1].bt;}
    for (int i = 0; i < n; i++) {
        proc[i].tat = proc[i].bt + proc[i].wt;   }
    printf("PID\tBurst Time\tWaiting Time\tTurnaround Time\n");
    int total_wt = 0, total_tat = 0;
    for (int i = 0; i < n; i++) {
        printf("%d\t%d\t\t%d\t\t%d\n", proc[i].pid, proc[i].bt, proc[i].wt, proc[i].tat);
        total_wt += proc[i].wt;
        total_tat += proc[i].tat;}
    printf("Average Waiting Time: %.2f\n", (float)total_wt / n);
    printf("Average Turnaround Time: %.2f\n", (float)total_tat / n);
    return 0;
}
```

Output

```
Enter number of processes: 6
Enter burst time for process 1: 2
Enter burst time for process 2: 36
Enter burst time for process 3:
8
Enter burst time for process 4: 9
Enter burst time for process 5: 7
Enter burst time for process 6: 8
PID Burst Time  Waiting Time    Turnaround Time
1   2       0       2
2   3       2       5
5   7       5       12
3   8       12      20
6   8       20      28
4   9       28      37
Average Waiting Time: 11.17
Average Turnaround Time: 17.33


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

main.c

Share Run

```c
#include <stdio.h>
typedef struct {
    int pid;
    int bt;
    int rem_bt;
} Process;
void roundRobin(Process proc[], int n, int tq) {
    int time = 0;
    int completed = 0;
    printf("PID\tBurst Time\tTurnaround Time\tWaiting Time\n");
    while (completed < n) {
        for (int i = 0; i < n; i++) {
            if (proc[i].rem_bt > 0) {
                if (proc[i].rem_bt > tq) {
                    proc[i].rem_bt -= tq;
                    time += tq;
                } else {
                    time += proc[i].rem_bt;
                    proc[i].rem_bt = 0;
                    completed++;
                    int turnaround_time = time;
                    int waiting_time = turnaround_time - proc[i].bt;
                    printf("%d\t%d\t\t%d\t\t%d\n", proc[i].pid, proc[i].bt, turnaround_time, waiting_time);
                } }}}}
int main() {
    int n, tq;
    printf("Enter number of processes: ");
    scanf("%d", &n);
    Process proc[n];
    for (int i = 0; i < n; i++) {
        proc[i].pid = i + 1;
        printf("Enter burst time for process %d: ", proc[i].pid);
        scanf("%d", &proc[i].bt);
        proc[i].rem_bt = proc[i].bt;}
    printf("Enter the time quantum: ");
    scanf("%d", &tq);
    roundRobin(proc, n, tq);
    return 0;
}
```

Output

```
Enter number of processes: 3
Enter burst time for process 1: 6
Enter burst time for process 2: 6
Enter burst time for process 3: 9
Enter the time quantum: 2
PID Burst Time  Turnaround Time Waiting Time
1   6           14             8
2   6           16             10
3   9           21             12


--- Code Execution Successful ---
```