```c
//round robin
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

typedef struct process {
    char pname[10];
    int bt;
    int at;
    int wt;
    int rt;
    int tat;
} p1;

int main() {
    p1 p[10], swap, result[10], e[10];
    float avg_tat = 0, avg_wt = 0, avg_rt = 0;
    int n, tq, i, j, k = 0, l = 0;
    int nextprocess = 0, total_burst = 0, realtime = 0;

    printf("\nENTER THE NUMBER OF PROCESS: ");
    scanf("%d", &n);

    printf("\nENTER THE TIME QUANTUM: ");
    scanf("%d", &tq);

    for (i = 0; i < n; i++) {
        sprintf(p[i].pname, "p%d", i + 1);
        p[i].rt = -1;
        p[i].wt = 0;
        p[i].tat = 0;

        printf("\nENTER THE BURST TIME FOR %s: ", p[i].pname);
        scanf("%d", &p[i].bt);

        printf("ENTER THE ARRIVAL TIME FOR %s: ", p[i].pname);
        scanf("%d", &p[i].at);

        total_burst += p[i].bt;
    }

    for (i = 0; i < n - 1; i++) {
        for (j = i + 1; j < n; j++) {
            if (p[j].at < p[i].at) {
                swap = p[i];
                p[i] = p[j];
                p[j] = swap;
            }
        }
    }

    for (i = 0; i < n; i++) {
        e[i] = p[i];
    }

    if (i != (n - 1)) {
        nextprocess = p[i + 1].at;
    } else {
        nextprocess = 10000;
    }
```

```
    while (total_burst > 0) {
        if (e[0].bt > tq) {
            e[0].wt += (realtime - e[0].tat);
            if (e[0].rt == -1) {
                e[0].rt = realtime;
            }
            realtime += tq;
            e[0].tat = realtime;
            e[0].bt -= tq;

            swap = e[0];
            for (j = 0; j < (n - 1); j++) {
                e[j] = e[j + 1];
            }
            e[n - 1] = swap;

        } else if (e[0].bt > 0) {
            realtime += e[0].bt;
            e[0].wt += (realtime - e[0].tat);
            e[0].tat = realtime - e[0].at;
            total_burst -= e[0].bt;
            e[0].bt = 0;

            result[k] = e[0];
            k++;

            for (j = 0; j < (n - 1); j++) {
                e[j] = e[j + 1];
            }
        }
    }

    for (i = 0; i < k; i++) {
        avg_tat += result[i].tat;
        avg_wt += result[i].wt;
        avg_rt += result[i].rt;
    }

    if (k > 0) {
        avg_wt /= k;
        avg_tat /= k;
        avg_rt /= k;
    } else {
        avg_wt = avg_tat = avg_rt = 0;
    }

    printf("\nAVERAGE WAITING TIME: %f", avg_wt);
    printf("\nAVERAGE TURNAROUND TIME: %f", avg_tat);
    printf("\nAVERAGE RESPONSE TIME: %f", avg_rt);

    return 0;
}
```