ROUND ROBIN:

```
#include <stdio.h>
struct Process {
    char name[10];
    int at, bt, wt, tt, remaining_bt;
};
int main() {
    int n, i, time = 0, quantum, done = 0;
    float total_wt = 0, total_tt = 0;
    printf("Enter number of processes: ");
    scanf("%d", &n);
    struct Process p[100];
    for (i = 0; i < n; i++) {
        printf("Enter name, arrival time, burst time for process %d: ", i+1);
        scanf("%s %d %d", p[i].name, &p[i].at, &p[i].bt);
        p[i].remaining_bt = p[i].bt;
        p[i].wt = p[i].tt = 0;
    }
    printf("Enter time quantum: ");
    scanf("%d", &quantum);
    while (done < n) {
        int flag = 0;
        for (i = 0; i < n; i++) {
            if (p[i].remaining_bt > 0) {
                flag = 1;
                if (p[i].remaining_bt <= quantum) {
                    time += p[i].remaining_bt;
                    p[i].wt = time - p[i].at - p[i].bt;
                    p[i].tt = p[i].wt + p[i].bt;
                    p[i].remaining_bt = 0;
                    done++;
                } else {
                    time += quantum;
                    p[i].remaining_bt -= quantum;
                }
            }
        }
        if (!flag) time++; // If no process is left to execute, increment time
    }

    printf("\nName\tAT\tBT\tWT\tTT\n");
    for (i = 0; i < n; i++) {
        printf("%s\t%d\t%d\t%d\t%d\n", p[i].name, p[i].at, p[i].bt, p[i].wt, p[i].tt);
        total_wt += p[i].wt;
```

```
        total_tt += p[i].tt;
    }
    printf("\nAvg WT = %.2f\nAvg TT = %.2f\n", total_wt/n, total_tt/n);
    return 0;
}
```

```
Enter number of processes: 3
Enter name, arrival time, burst time for process 1: A
2
3
Enter name, arrival time, burst time for process 2: B
4
1
Enter name, arrival time, burst time for process 3: C
5

4
Enter time quantum: 2

Name    AT  BT  WT  TT
A   2   3   1   4
B   4   1   -2  -1
C   5   4   -1  3

Avg WT = -0.67
```