#include <stdio.h>

int main() {

int cnt, j, n, t, remain, flag = 0, tq;

int wt = 0, tat = 0; // Waiting Time and Turnaround Time totals

int at[10], bt[10], rt[10]; // Arrival Time, Burst Time, Remaining Time

printf("Enter Total Number of Processes: ");

scanf("%d", &n);

remain = n;

// Input Arrival Time and Burst Time for each process

for (cnt = 0; cnt < n; cnt++) {

printf("Enter Arrival Time and Burst Time for Process %d:\n", cnt + 1);

printf("Arrival Time: ");

scanf("%d", &at[cnt]);

printf("Burst Time: ");

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

rt[cnt] = bt[cnt]; // Initialize Remaining Time

}

// Input Time Quantum

printf("Enter Time Quantum: ");

scanf("%d", &tq);

printf("\n\nProcess\t| Turnaround Time | Waiting Time\n\n");

// Round Robin Scheduling Logic

for (t = 0, cnt = 0; remain != 0;) {

if (rt[cnt] <= tq && rt[cnt] > 0) {

t += rt[cnt]; // Add remaining burst time to the current time

rt[cnt] = 0; // Process finished

flag = 1;

} else if (rt[cnt] > 0) {

rt[cnt] -= tq; // Subtract time quantum from remaining burst time

t += tq; // Increment time by time quantum

}

if (rt[cnt] == 0 && flag == 1) { // Process completed

remain--;

printf("P[%d]\t|\t%d\t|\t%d\n", cnt + 1, t - at[cnt], t - at[cnt] - bt[cnt]);

wt += t - at[cnt] - bt[cnt]; // Add Waiting Time

tat += t - at[cnt]; // Add Turnaround Time

flag = 0;

}

// Move to the next process in round-robin

if (cnt == n - 1)

cnt = 0;

else if (at[cnt + 1] <= t)

cnt++;

else

cnt = 0;

}

// Calculate and display averages

printf("\nAverage Waiting Time = %.2f\n", wt \* 1.0 / n);

printf("Average Turnaround Time = %.2f\n", tat \* 1.0 / n);

return 0;

}