#include <bits/stdc++.h>

using namespace std;

struct data{

int id;

int arrivalTime;

int burstTime; // to update value

int completionTime;

int turnAroundTime;

int originalBurstTime;

int waitingTime;

bool isCompleted;

};

void shortestJobFirst(struct data d[], int n, int timeQuantum){

int currentTime = 0;

while (true){

int idOfProcess = -1;

int minBurstTime = 99999;

for (int i = 0; i < n; i++){

if (d[i].arrivalTime <= currentTime && d[i].isCompleted == false && d[i].burstTime < minBurstTime){

idOfProcess = d[i].id;

minBurstTime = d[i].burstTime;

}

}

// To check whether all process completed

bool processRemaining = false;

for(int i = 0;i<n;i++) if(d[i].isCompleted == false) processRemaining = true;

if(processRemaining == false) break;

if (d[idOfProcess].burstTime - timeQuantum <= 0){

currentTime = currentTime + d[idOfProcess].burstTime;

d[idOfProcess].burstTime = 0;

d[idOfProcess].isCompleted = true;

d[idOfProcess].completionTime = currentTime;

d[idOfProcess].turnAroundTime = d[idOfProcess].completionTime - d[idOfProcess].arrivalTime;

d[idOfProcess].waitingTime = d[idOfProcess].turnAroundTime - d[idOfProcess].originalBurstTime;

} // Else execute process and update burst time

else if(idOfProcess == -1){ // If no process is there and cpu is in idle state

currentTime += timeQuantum;

}

else{

currentTime += timeQuantum;

d[idOfProcess].burstTime -= timeQuantum;

}

}

cout << endl;

cout << "Id "

<< "AT "

<< "BT "

<< "CT "

<< "TAT "

<< "WT "

<< endl;

float totalTAT = 0;

float totalWT = 0;

for (int i = 0; i < n; i++){

totalTAT += d[i].turnAroundTime;

totalWT += d[i].waitingTime;

cout << d[i].id + 1 << " " << d[i].arrivalTime << " " << d[i].originalBurstTime << " " << d[i].completionTime<<" "<<d[i].turnAroundTime<<" "<<d[i].waitingTime;

cout << endl;

}

printf("Average Waiting Time = %f",totalTAT/n);

printf("\nAverage Turn Around Time = %f",totalWT/n);

}

int main()

{

int n;

int timeQuantum;

cout << "Enter the no of processes = ";

cin >> n;

cout << "Enter the time quantum = ";

cin >> timeQuantum;

struct data d[n];

for (int i = 0; i < n; i++)

{

d[i].id = i;

d[i].isCompleted = false;

printf("\nEnter Arrival Time:");

scanf("%d", &d[i].arrivalTime);

printf("Enter Burst Time:");

scanf("%d", &d[i].originalBurstTime);

d[i].burstTime = d[i].originalBurstTime;

}

shortestJobFirst(d, n, timeQuantum);

}