IEEE Women In Engineering Problem

IEEE Women In Engineering mission is to facilitate the global recruitment and retention of women in technical disciplines. To do that, they decided to release a problem for public audience to optimize city train system as a gesture that everyone can be an Engineer. The problem as follows:

For m trains and single track with n stations, compute station arrival and departure times so that only zero or one train is between any pair of stations at any time and inter-station travel time is minimized. Given: acceleration rate, maximum speed, deceleration rate, minimum time stopped at a station and lengths of track, sections, trains, and stations.

Description

Trains leave a depot and arrive at a terminal, running over a single track. The track is 100 kilometers long. A track may have from 0 to 10 stations. The track between two stations is called a “section” and is at least 500 meters. Each station parallels 150 meters of track. Figure 1 shows these relationships.

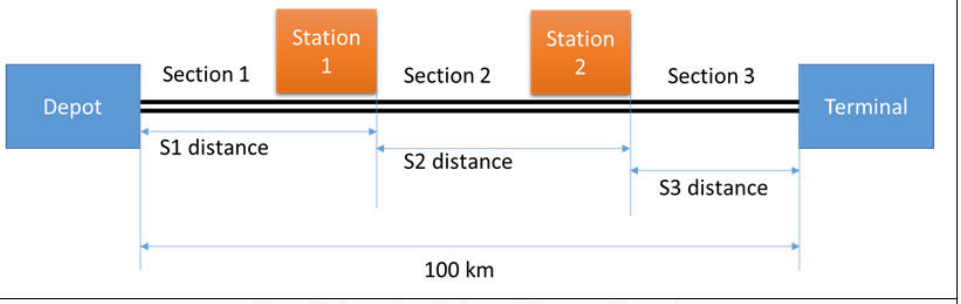

Figure 1: Example with two stations, not to scale

A train is 100 meters long. All trains accelerate at constant rate of 2.7 kilometers per hour per second, cruise at a top speed of 90 kilometers per hour, and decelerate at a constant rate of 3.8 kilometers per hour per second. Trains must accelerate, cruise, and decelerate so that they to come to a complete stop at the next station while minimizing the time to travel a section.

Recall that d=(a*t^2)/2, where distance traveled d is a function of acceleration (or deceleration) a over period t, assuming an initial velocity of 0. In short sections, the train will not reach its cruising speed—it will simply accelerate then decelerate. For a sufficiently long section, the train accelerates to cruising speed, cruises, and then decelerates to stop at the station.

A train must stop at a station for at least two minutes after it arrives. Trains stop at the far edge of the station. A train is considered to have arrived at the Terminal when the first car reaches the Terminal. To ensure safety, only one train may be in any section at time. If train is ready to depart, it must wait until the next station is empty. A train may depart one second after the next station becomes empty. The last section is considered clear when a train arrives at the Terminal.

Task

For a given set of sections up to five trains, write a program that computes the arrival and departure time for each train at each station, achieving all of the stated operating conditions.

Input

The program must accept input in the form N {d} 0..m, for N trains, section distance dm, or , <section 1 distance >, < section 2 distance>,…, < section m distance >

All inputs are unsigned integers, separated with a space. All distances are in meters
- Accept from 1 to 5 trains - Accept from 1 to 5 sections - The total of the sections must equal 100000

Output Data

If the input format is invalid or conflicts with the problem specification, the program must output only “ERROR” The schedule is of the form {n:{ta-td}}

Where n is the train number followed by a colon, ta is the arrival time at the mth station and td is the departure time at station m+1. Arrival-departure times are separated with a dash.

Schedule times for the Depot and Terminal are special cases: the arrival and departure times are the same. The first train departs the Depot at time 1. All output values are unsigned integers. Times are in seconds, rounded to the nearest whole second.

The output format must follow the below format. The tilde indicates a single space. Assuming one train T, and two sections, the output would be:

```
T~: *****~–~ddddd~~aaaaa~–~ddddd~~aaaaa~*****
```

Where ddddd is the departure time, aaaaa is the arrival time. Both are five digits with leading zeros suppressed. The asterisks indicate an unspecified arrival or departure time. See following examples.

Sample Input1:

```
3 400 400 99000
```

Sample Output1:

```
ERROR
```

Sample Input2:

```
0 50000 50000
```

Sample Output2:

```
ERROR
```

Sample Input3:

```
2 60000 60000
```

Sample Output3:

```
ERROR
```

Sample Input4:

```
12 10000 10000 10000 10000 10000 10000 10000 10000 10000 10000 10000 10000
```

Sample Output 4:

```
ERROR
```

Sample Input5:

```
2 5000 50000
```

Sample Output 5:

```
ERROR
```

Sample Input6:

```
Fja 3 nasdfpij NASD;ASD
```

Sample Output 6:

```
ERROR
```

Sample Input 7 (One train, one section):

```
1 100000
```

Sample Output 7:

```
1 : ***** -     1   4030 *****
```

Sample Input 8 (Two trains, three sections)

```
2 500 500 99000
```

Sample Output 8:

```
1 : ***** -     1    49 -   170    218 -   339   4328 *****
2 : ***** -   171   219 -   340    388 -  4329   8318 *****
```

Problem Author: IEEE

Suggest Edits

Emacs | Normal | Vim

Select Language: Java

save code

```
1  import java.io.*;
2  import java.util.*;
3  import java.text.*;
4  import java.math.*;
5  import java.util.regex.*;
6
7  public class Solution {
8
9      public static void main(String[] args) {
10         /* Enter your code here. Read input from STDIN. Print output to STDOUT. Your class
   should be named Solution. */
11     }
12 }
```

Line: 12 Col: 2 Count: 302

Use a custom test case

Upload Code as File

Compile & Test | Submit Code