# JS Advanced: Exam 26 April 2018

Problems for exam preparation for the [“JavaScript Advanced” course @ SoftUni](https://softuni.bg/courses/javascript-advanced). Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/1004/>.

# Problem 3. Line Manager (Simple Class)

Write a JavaScript class LineManager that keeps information about the course of a bus. The class holds a collection of **stops**, the **current** **stop**, and the **duration** of time that the bus has traveled.

|  |
| --- |
| **class** LineManager {  *//* ***TODO: implement this class*** } |

The class **constructor** should receive an array of **stops** (see below for details).

Implement the following features:

Each **stop** in the array is an **object** that contains a **name** (string) and **time** traveled to next stop in minutes (number) it has the following format:

{

name: String,

timeToNext: Number

}

Also each stop should be validated. Name should be a non-empty string and time should be a positive number (zero is included). In case of an invalid stop throw an Error with an appropriate message.

Getter atDepot – returns true if the current stop is the last stop, otherwise returns false.

Getter nextStopName – returns the name of the next stop. If the bus is at the last stop return the string “At depot.”

Getter currentDelay – returns the delay in minutes that a bus has made during the entire trip (check the example for details).

Function arriveAtStop(minutes) – receives a minutes **parameter** that should be **validated**. If the parameter is a **negative number** or the bus is at **depot (no more stops left)** **throw an Error** with an appropriate message. The function should **add** the duration of minutes and **change** the current stop to the next one. It should return **true** if the current stop is **not** the last stop, otherwise it returns **false**.

Function toString() – return a string, containing a summary about the **current situation** of a bus (see examples for formatting details)

***Scroll down for examples and constraints.***

### Examples

|  |
| --- |
| Sample code usage |
| *// Initialize a line manager with correct values* **const *man*** = **new** LineManager([  {**name**: **'Depot'**, **timeToNext**: 4},  {**name**: **'Romanian Embassy'**, **timeToNext**: 2},  {**name**: **'TV Tower'**, **timeToNext**: 3},  {**name**: **'Interpred'**, **timeToNext**: 4},  {**name**: **'Dianabad'**, **timeToNext**: 2},  {**name**: **'Depot'**, **timeToNext**: 0}, ]);  *// Travel through all the stops until the bus is at depot* **while**(***man***.atDepot === **false**) {  **console**.log(***man***.toString());  ***man***.arriveAtStop(4); }  **console**.log(***man***.toString());  *// Should throw an Error (minutes cannot be negative)* ***man***.arriveAtStop(-4); *// Should throw an Error (last stop reached)* ***man***.arriveAtStop(4);  *// Should throw an Error at initialization* **const *wrong*** = **new** LineManager([  { **name**: **'Stop'**, **timeToNext**: { **wrong**: **'Should be a number'**} } ]); |
| Corresponding output |
| Line summary  - Next stop: Romanian Embassy  - Stops covered: 0  - Time on course: 0 minutes  - Delay: 0 minutes  Line summary  - Next stop: TV Tower  - Stops covered: 1  - Time on course: 4 minutes  - Delay: 0 minutes  Line summary  - Next stop: Interpred  - Stops covered: 2  - Time on course: 8 minutes  - Delay: 2 minutes  Line summary  - Next stop: Dianabad  - Stops covered: 3  - Time on course: 12 minutes  - Delay: 3 minutes  Line summary  - Next stop: Depot  - Stops covered: 4  - Time on course: 16 minutes  - Delay: 3 minutes  Line summary  - Course completed  - Stops covered: 5  - Time on course: 20 minutes  - Delay: 5 minutes |

### Constraints

* Your class will be tested with both **valid and invalid parameters** and should validate the input to the constructor and arriveAtStop.

### Submission

Submit **only** your class LineManager.

### Hint

To create a string, that contains a line break, use the special character '\n'.