# JS Advanced Retake - 08 August 2019

Exam problems for the ["JavaScript Advanced" course @ SoftUni](https://softuni.bg/courses/js-advanced). Submit your solutions in the SoftUni Judge system at <https://judge.softuni.bg/Contests/Practice/Index/1772#2>.

# Problem 3. Library

**class** Library {  
 *//* ***TODO: implement this class...***  
}

### Your Task

### Write a Library class which supports the described functionality below.

### Functionality

#### constructor()

Receives **1** parameters at initialization of the class (**libraryName**), where library name is a **string.**

Should have at least these **3** properties:

* **libraryName** - **string** (should be the same as the received **libraryName**)
* **subscribers** - **empty** **array**
* **subscrptionTypes** - object with properties **normal, special** and **vip**
  + The number of books that a person with **normal** subscription can receive is equal to the **length of the libraryName**
  + The number of books that a person with **special** subscription can receive is equal to the **length of the libraryName multiplied by 2 (libraryName \* 2)**
  + A person with **vip** subscription can receive unlimited(**Number.MAX\_SAFE\_INTEGER**) amount of books

#### subscribe(name, type)

This **function** receives **2 string** parameters - **name** and **type**

* If the given subscription **type** is not **normal**, **special** or **vip**, a new error should be **thrown** with the following message: **"The type {type} is invalid"**
* If the person **is not subscribed**, you should make **new** **subscriber** **object** with properties
  + **name** (the subscriber name)
  + **type** (the subscription type)
  + **books** (an empty array by default)

and add it to the library subscribers' array.

* If there is a person with that **name** in the **subscribers** list, you should just **change** his subscription **type** with the given type.

This function should **return** the current **subscriber**.

#### unsubscribe(name)

This **function** receives **1** parameter **name** and should **unsubscribe** an **already** **subscribed** person in the library (**remove** the person with the **given** **name** from the **subscriber's** **property**).

* If there **is no subscriber** with that **name,** a **new error** should be **thrown** with the following message: "**There is no such subscriber as {name}**"
* If **subscribers** **property** contains a person with the **given** **name**, that person should be **removed** from the array.

This function should **return** the library's **subscribers** list

#### receiveBook(subscriberName, bookTitle, bookAuthor)

This function receives **3** parameters **(subscriberName, bookTitle** and **bookAuthor**) and should **add** a book to the **subscriber's** book list.

* If there is **no** such subscriber in the **subscriber's** array, a new error should be **thrown** with the following message: **"There is no such subscriber as {name}"**
* If there is a subscriber with that name you should **check** his subscription **type**:
  + If his subscription type **allows** him to **receive** more book you should **add** a new **book** **object** with properties **title** and **author** to his books array
  + Otherwise a new error should be thrown, with the following message:

**"You have reached your subscription limit {subTypeLimit}!"**

This function should return the **subscriber** with the given name.

#### showInfo ()

This function should **return a string with all the subscribers** with their books joined by (**", "**) in the following format:

**"Subscriber: {subscriberName}, Type: {subscriptionType}\n**

**Received books: {title} by {author}, {title2} by {author2}…"**

If the subscriber's property in the Library is empty just **return** the following string:

**"{libraryName} has no information about any subscribers"**

### Submission

Submit only your **Library class.**

### Examples

This is an example how the code is **intended to be used**:

|  |
| --- |
| Sample code usage |
| let **lib** **=** **new** Library(**'Lib'**);  **lib.**subscribe(**'Peter'**, **'normal'**);  **lib.**subscribe(**'John'**, **'special'**);  **lib.**receiveBook(**'John'**, **'A Song of Ice and Fire'**, **'George R. R. Martin'**);  **lib.**receiveBook(**'Peter'**, **'Lord of the rings'**, **'J. R. R. Tolkien'**);  **lib.**receiveBook(**'John'**, **'Harry Potter'**, **'J. K. Rowling'**);  console**.**log(**lib.**showInfo()); |
| Corresponding output |
| Subscriber: Peter, Type: normal  Received books: Lord of the rings by J. R. R. Tolkien  Subscriber: John, Type: special  Received books: A Song of Ice and Fire by George R. R. Martin, Harry Potter by  J. K. Rowling |

*GOOD LUCK!😊*