1 Practice 5.1. Classes

> This practice is the next in the series of practices consisting of a Contacts application. This practice is based on the previous one. Before start it, finish the previous one, make the corrections you consider, if needed, and start coding.
>
> The inclusion of comments in the scripts will be valued.
>
> Before start, create a new git branch named `contacts-u5` and change to it:
>
> ```
> git branch contacts-u5
> git checkout contacts-u5
> ```
>
> Do all the practice in the contacts-u5 branch. Once finished, do the submission via GitHub Classroom as detailed in the end of the practice.

1.1 Exercise 1

Make a class, named **`Contact`**, for your contacts. The class must have the next **private** properties:

- $id (int)
- $title (string)
- $name (string)
- $surname (string)
- $birthdate (string)
- $phone (string)
- $email (string)
- $favourite (bool)
- $important (bool)
- $archived (bool)

Each property must have its **set** and **get** methods. These methods must be **strictly typed**.

Add a **constructor** that gets an optional array as parameter and assigns the values of the array to the object properties. If no parameters are passed, assign them default values.

For example, the constructor definition could be:

```
public function __construct(array $contactArray = array("id"=> "", "title"=>
↪   "Mr.", "name"=> "", "surname"=> "", "birthdate"=>"", "phone"=>"",
↪   "email"=>"", "favourite"=>false, "important"=>false,
↪   "archived"=>false))
```

Override the method **`__toString()`**, to return a string with all the data of the object, formatted properly.

Move the functions `checkContactDate` and `checkBirthday` from the file `functions.php` from the Unit 3 to the class as methods. Their signatures should be:

- **`checkContactDate() : bool`**
- **`checkBirthday() : ?int`**

Modify your `contact_form.php` script to work with this class for, in addition to the previous validations and functionality, doing the next with the methods:

- Use the `checkContactDate` function to check the validity of the date in the `contact_form` script.
- Use the `checkBirthday` function to calculate the numbers of days until the contact's birthday in the `contact_form` script. If the number of days returned is 0, show the message:
  - *Today is CONTACTNAME's birthday.* (replace CONTACTNAME with the contact's name).

  If the number of days is between 1 and 7, show the message:
  - *There are X days left for CONTACTNAME's birthday.*

1.2 Exercise 2

Make a class, named **`ContactList`**. This class must store a list of contacts of the **Contact** class. The class must have the next properties:

- $contactList (array of Contact objects).

Create the public methods:

- **`getContactList(): array`**: returns an array of Contact objects.
- **`addContact(Contact $contact): void`**: adds a Contact object to the Contact list.
- **`addContactList(array $contacts): void`**: adds an array of Contact objects to the Contact list,
- **`getContactById(int $id): ?Contact`**: returns a Contact from the Contact list by its id. If not found, must return a **`null`** value (the ? before the return type means that it must be of type Contact or null).

These methods must be **strictly typed**.

Modify your `contact_list.php` script to work with the classes `Contact` and `ContactList`. Instead of handling the array `$contacts` from the `data.php` file, delete the file and create a `ContactList` object with all the elements from the array and retrieve all the contacts from it using its public methods.

1.3 How to submit to GitHub Classroom

1. This task must be submitted to the same repository of the previous one (Practice 4.1). Remember to make a new branch before start the practice.
2. Once you finish the task, make a commit with the comment "PRACTICE 5.1 SUBMISSION COMMIT" and push it to GitHub. Before that, you can do the commits and pushes you want. If you change your code after your submission commit, make another commit and push with the same text in the message adding the corrections you've done.Make a pull-request if you want so i could know your code is submitted.
3. Once pushed to GitHub, create a tag with the name "Practice 5". Create a new tag if you make corrections after the first submission (ex. "Practice 5 corrected").

If you have any doubt in your task, you can push your code and ask me by email what's your problem.