Software testing

Unit tests in Java using JUnit - Exercise solutions

> **Exercise 2:**
>
> Implement a class called *Access* with a method called *validUser(String user,String pass)* that returns *true* if user is valid and *false* if it is not. In order for a user to be valid, it must meet the following conditions:
>
> - *user* parameter must start with a letter.
> - *user* parameter must have a length between 7 and 10 characters.
> - Password must have a minimum length of 10 characters, and it must have at least one letter and one number.
>
> Design the test cases using the technique of the equivalent partition, and then implement these test cases with JUnit.

Let's go with the test case design. In this case, we have these equivalence classes for each parameter:

Parameter	Valid class	Invalid class
user	Start with letter AND have between 7 and 10 character	Start something which is NOT a letter OR have less than 7 characters OR have more than 10 characters
password	Have at least 10 characters AND have at least one letter AND have at least one digit	Have less than 10 characters OR do not have any letter OR do not have any digit

According to these equivalence classes, we can build this table for the test cases:

ID	Name	Steps	Data	Expected result
U1	ValidData	Enter valid data for user and password	user="pepito123", password "sanvicente2019"	true
U2	InvalidUser1	Enter user not starting with letter	user="123pepito", password="sanvicente2019"	false
U3	InvalidUser2	Enter user shorter than 7 characters	user="pepe12", password="sanvicente2019"	false
U4	InvalidUser3	Enter user greater than 10 characters	user="pepito12345", password="sanvicente2019"	false
U5	InvalidPassword1	Enter password shorter than 10 characters	user="pepito123", password="sanvi19"	false
U6	InvalidPassword2	Enter password with at least 10 characters, not containing any letter	user="pepito123", password="123123123123"	false
U7	InvalidPassword3	Enter password with at least 10 characters, not containing any digit	user="pepito123", password="aabbccddeeff"	false
U8	InvalidUser4	Enter a null user	user=null, password="sanvicente2019"	false
U9	InvalidPassword4	Enter a null password	user="pepito123", password=null	false

Regarding the implementation of these test cases, you have it in the *Access* project next to this document

> **Exercise 3:**
>
> Add a new method to the *Access* class: *boolean register(String user, String pass)* that will register the user. The registration will consist in adding the user to a map. There will be a maximum of 10 allowed users in the map, so that if we exceed this limit, the method will return *false*. If the registration is correct, it will return *true*. Also, if a user with the existing name already exists in the map, it will return *false*. You are also asked to implement the tests in JUnit. In order to check if we exceed the limit of the map, we need to implement a method with the *@BeforeEach* annotation that adds 10 users to the map, so that when we invoke the corresponding test, it will return *false*.

In this case, the test cases are simple:

- Try to register a valid, non-existing user (result `true`)
- Try to register an invalidad, already existing user (result `false`)

- Try to register a user that exceeds the map limit. In order to do this, if we are not using JUnit 5, we can't rely on `@BeforeEach` annotation, so we just need to fill the map with 10 users and test if trying to register one more user returns `false`.

The implementation of this method and its corresponding unit test are also included in the `Access` project, in the ZIP file next to this document.