# Exercises: Unit Testing

This document defines the exercises for ["C# OOP Advanced" course @ Software University](https://softuni.bg/trainings/2085/csharp-oop-advanced-november-2018).

## Database

Create a simple class - **Database**. It should **store integers**. You should **set the initial integers by constructor**. Store them **in an array**. Your Database should have a functionality to **add**, **remove** and **fetch all of the stored items**. Your task is to **test the class**. In other words, create the class and **write tests** to ensure its methods are working as intended.

### Constraints

* Storing array's **capacity** must be **exactly 16 integers**
  + If the size of the array is not 16 integers long, throw **InvalidOperationException**
* **Add** operation, should **add an element at the next free cell** (just like a stack)
  + If there are 16 elements in the Database and try to add 17th, throw an **InvalidOperationException**
* **Remove** operation, should support only removing the element **at the last index** (just like a stack)
  + If you try to remove an element from an empty Database throw **InvalidOperationException**
* **Constructors** should take integers only, and store them in **an** **array**
* **Fetch method** should return the elements as **array**

### Hint

Do not forget to **test the constructor(s)**. They are methods too!

## Extended Database

You already have a class - **Database**. Now your task is to modify and extend it. It should support, **adding, removing and finding People**. In other words, it should **store People**. There should be two types of finding methods - first: **FindById (long id)** and the second one: **FindByUsername (string username)**. As you may have already guessed, each person should have its own **unique id**, and **unique username**. Your task is to implement these functions and test them.

### Constraints

Database should have methods:

* Add
  + If there are already users with this username, throw **InvalidOperationException**
  + If there are already users with this id, throw **InvalidOperationException**
* Remove
* FindByUsername
  + If no user is present by this username, throw **InvalidOperationException**
  + If username parameter is null, throw **ArgumentNullException**
  + Arguments are all **CaseSensitive**
* FindById
  + If no user is present by this id, throw **InvalidOperationException**
  + If negative ids are found, throw **ArgumentOutOfRangeException**

### Hint

Do not forget to test the constructor(s). They are methods too!

## Custom Linked List

Use the VS **solution** "**CustomLinkedList**".

* Create new **Unit** **Test** **Project** and **add** **reference** to the “**CustomLinkedList**”.
* Create Test Methods for **all public members** that need testing.
* Create tests that ensure all methods, getters and setters **work correctly** (do not test auto-properties).
* Make sure that the methods throw the correct exceptions in case a wrong input is entered.
* Give **meaningful** **assert** **messages** for failed tests.

## Storage Master

You are given a quite familiar C# OOP Basic Exam - Storage Master. You have been provided with the author’s solution, and your task is to create unit tests for the skeleton structure and for the business logic.

* For the **StorageMester.Tests.Structure** you need to test if all fields, consts, propeties, constructors and methods exists.
* For the **StorageMester.BusinessLogic.Tests** you need to test if all business methods are implemented properly.