# Task description

You are given an already built software system. Your task is to get to know with the way the system is working. Check out the tests requirements, to know exactly what you must test. For a warm-up - start with the tests that look trivial to you (those that you know you can implement in seconds). Identify weird test cases. Try solving them by yourself. If you are having troubles or questions - ask any of the trainers, or any of your fellow colleagues for help.

## Tests Requirements

### Class - Cosmetics.Common.Validator

#### Test cases:

- \*\*CheckIfNull\*\* should throw \*\*NullReferenceException\*\*, when the parameter \*\*"obj"\*\* is null.

- \*\*CheckIfNull\*\* should \*\*NOT throw\*\* any Exceptions, when the parameter \*\*"obj"\*\* is NOT null.

- \*\*CheckIfStringIsNullOrEmpty\*\* should throw \*\*NullReferenceException\*\*, when the parameter \*\*"text"\*\* is null or empty.

- \*\*CheckIfStringIsNullOrEmpty\*\* should \*\*NOT throw\*\* any Exceptions, when the parameter \*\*"text"\*\* is NOT null or empty.

- \*\*CheckIfStringLengthIsValid\*\* should throw \*\*IndexOutOfRangeException\*\*, when the length of the parameter \*\*"text"\*\* is lower than the minimum allowed value or greater than the maximum allowed value.

- \*\*CheckIfStringLengthIsValid\*\* should \*\*NOT throw\*\* any Exceptions, when the length of the parameter "text" is in the allowed boundaries.

### Class - Cosmetics.Engine.Command

#### Test cases:

- \*\*Parse\*\* should \*\*return new Command\*\*, when the "input" string is in the required valid format.

- \*\*Parse\*\* should set correct values to the newly returned Command object's Properties ("Name" & "Parameters"), when the "input" string is in the valid required format.

- \*\*Parse\*\* should throw \*\*ArgumentNullException\*\* with a message that contains the string "Name", when the "input" string that represents the Command Name is Null Or Empty.

- \*\*Parse\*\* should throw \*\*ArgumentNullException\*\* with a message that contains the string "List", when the "input" string that represents the Command Parameters is Null or Empty.

### Class - Cosmetics.Engine.CosmeticsEngine

#### Test cases:

- \*\*Start\*\* should throw \*\*ArgumentNullException\*\*, when the "input" string of commands is not in the correct format.

- \*\*Start\*\* should read, parse and execute \*\*"CreateCategory" command\*\*, when the passed input string is in the format that represents a CreateCategory command, which should result in adding the new Category in the list of categories.

- \*\*Start\*\* should read, parse and execute \*\*"AddToCategory" command\*\*, when the passed input string is in the format that represents a AddToCategory command, which should result in adding the selected product in the respective category.

- \*\*Start\*\* should read, parse and execute \*\*"RemoveFromCategory" command\*\*, when the passed input string is in the format that represents a RemoveFromCategory command, which should result in removing the selected product from the respective category.

- \*\*Start\*\* should read, parse and execute \*\*"ShowCategory" command\*\*, when the passed input string is in the format that represents a ShowCategory command, which should result in calling the Print() method of the respective cateogry.

- \*\*Start\*\* should read, parse and execute \*\*"CreateShampoo" command\*\*, when the passed input string is in the format that represents a CreateShampoo command, which should result in adding the newly created shampoo in the list of products.

- \*\*Start\*\* should read, parse and execute \*\*"CreateToothpaste" command\*\*, when the passed input string is in the format that represents a CreateToothpaste command, which should result in adding the newly created toothpaste in the list of products.

- \*\*Start\*\* should read, parse and execute \*\*"AddToShoppingCart" command\*\*, when the passed input string is in the format that represents a AddToShoppingCart command, which should result in adding the selected product to the shopping cart.

- \*\*Start\*\* should read, parse and execute \*\*"RemoveFromShoppingCart" command\*\*, when the passed input string is in the format that represents a RemoveFromShoppingCart command, which should result in removing the selected product from the shopping cart.

### Class - Cosmetics.Engine.CosmeticsFactory

#### Test cases:

- \*\*CreateShampoo\*\* should throw \*\*ArgumentNullException\*\*, when the passed "name" parameter is invalid. (Null or Empty, or with length out of range)

- \*\*CreateShampoo\*\* should throw \*\*ArgumentNullException\*\*, when the passed "brand" parameter is invalid. (Null or Empty, or with length out of range)

- \*\*CreateShampoo\*\* should return a \*\*new Shampoo\*\*, when the passed parameters are all valid.

- \*\*CreateCategory\*\* should throw \*\*ArgumentNullException\*\*, when the passed "name" parameter is invalid. (Null or Empty, or with length out of range)

- \*\*CreateCategory\*\* should return a \*\*new Category\*\*, when the passed parameters are all valid.

- \*\*CreateToothpaste\*\* should throw \*\*ArgumentNullException\*\*, when the passed "name" parameter is invalid. (Null or Empty, or with length out of range)

- \*\*CreateToothpaste\*\* should throw \*\*ArgumentNullException\*\*, when the passed "brand" parameter is invalid. (Null or Empty, or with length out of range)

- \*\*CreateToothpaste\*\* should throw \*\*IndexOutOfRangeException\*\*, when the count of items in the list of ingredients is not in the allowed boundaries.

- \*\*CreateShoppingCart\*\* should always return a new \*\*ShoppingCart\*\*.

### Class - Cosmetics.Products.Shampoo/Toothpaste/Category/ShoppingCart

#### Test cases:

- \*\*Shampoo.Print()\*\* should return a string with the shampoo details in the required format.

- \*\*Toothpaste.Print()\*\* should return a string with the toothpaste details in the required format.

- \*\*Category.Print()\*\* should return a string with the category details in the required format.

- \*\*ShoppingCart.Print()\*\* should return a string with the shopping cart details in the required format.