Lab: Generics

You can check your solutions here: https://judge.softuni.bg/Contests/3183/Additional-Exercises.

Part I: Generics

1. Box of T

NOTE: You need a public **StartUp** class with the namespace **BoxOfT**.

Create a class **Box<>** that can store anything. It should have two public methods:

- **void Add(element)** – adds an element on the top of the list.
- **element Remove()** – removes the topmost element.
- **int Count { get; }**

Examples

```
public static void Main()
{
    Box<int> box = new Box<int>();
    box.Add(1);
    box.Add(2);
    box.Add(3);
    Console.WriteLine(box.Remove());
    box.Add(4);
    box.Add(5);
    Console.WriteLine(box.Remove());
}
```

Hints

Use the syntax **Box<T>** to create a generic class

2. Generic Array Creator

NOTE: You need a public **StartUp** class with the namespace **GenericArrayCreator**.

Create a class **ArrayCreator** with a method and a single overload to it:

- **static T[] Create(int length, T item)**

The method should return an array with the given length and every element should be set to the given default item.

Examples

```
static void Main(string[] args)
{
   string[] strings = ArrayCreator.Create(5, "Pesho");
   int[] integers = ArrayCreator.Create(10, 33);
}
```

Part II: Generic Constraints

3. Generic Scale

NOTE: You need a public **StartUp** class with the namespace **GenericScale**.

Create a class **EqualityScale<T>** that holds two elements - **left** and **right**. The scale should receive the elements through its single constructor:

- **`EqualityScale(T left, T right)`**

The scale should have a single method:

- **`bool AreEqual()`**

The greater of the two elements is the heavier. The method should return **`true`** if the elements are equal.

Part III: Change Solution to Generic

4. Make Your CustomArrayList Class Generic

Use the class **`CustomArrayList`** from the solution of the Implementing Array List lesson and modify your code to **use generics**.

Use the **`default`** keyword, it returns the default value of type parameter. For example:

```
arr[Count - 1] = null;
```

Will become:

```
arr[Count - 1] = default(T);
```

Hints

This is how your generic class **`CustomArrayList<T>`** may look like:

```
namespace ImplementArrayList
{
    2 references
    public class CustomArrayList<T>
    {
        private T[] arr;
        private static int INITIAL_CAPACITY = 4;
        private int count;
```