# Lab: Generics

Problems for exercises and homework for the ["Java OOP Advanced" course @ SoftUni](https://softuni.bg/courses/java-oop-advanced).

You can check your solutions here: <https://judge.softuni.bg/Contests/Practice/Index/521#0>.

# Part I: Generics

## Jar of T

Create a class **Jar<>** that can store anything.

It should have two public methods:

* void add(element)
* element remove()

Adding should add on top of its contents. Remove should get the topmost element.

### Examples



### Hints

Use the syntax Jar<T> to create a generic class

## Generic Array Creator

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

* static T[] create(int length, T item)
* static T[] create(Class<T>, 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



# Part II: Type Parameter Bounds

## Generic Scale

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

* Scale(T left, T right)

The scale should have a single method:

* T getHeavier()

The greater of the two elements is heavier. The method should return null if elements are equal.

### Examples



## List Utilities

Create a class ListUtils that you will use through several other exercises:

The class should have two static methods:

* T getMin(List<T> list)
* T getMax(List<T> list)

The methods should throw IllegalArgumentException if an empty list is passed.

### Examples



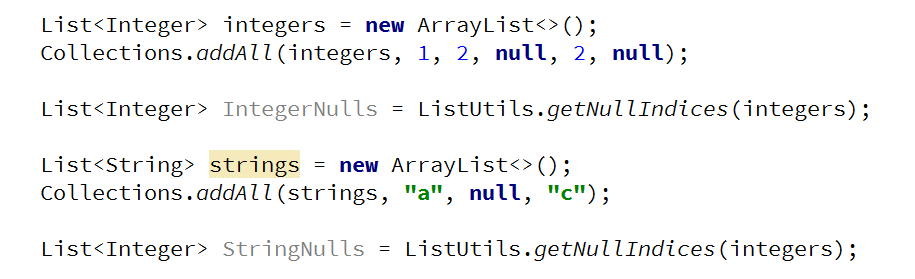
## Null Finder

Add a method to your ListUtils class that finds the index of every null element in a given list with the method:

* static List<Integer> getNullIndices(List<> list)

Add the appropriate **generic syntax** to the signature. The method should work with any List<>.

### Examples



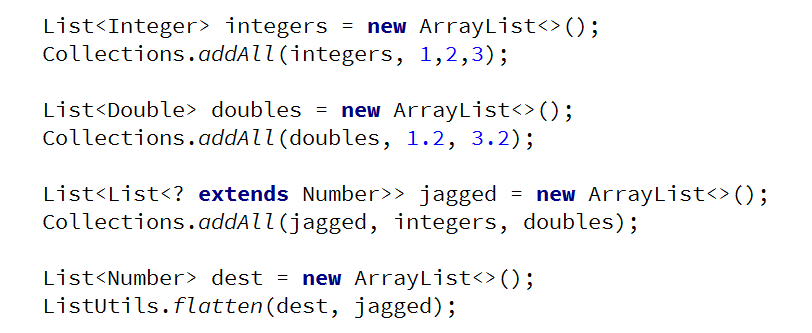
## Generic Flat Method

In ListUtils, create a generic static method that flattens a List<List<>> into a resulting List<>

* void flatten(List<> destination, List<List<>> source)

Add the appropriate generic syntax to the signature. The method should work with any List<>.

### Examples



## Generic Add All Method

In ListUtils, create a generic static method that **adds all elements from a given source** list **to a given destination** list with the static method:

* void addAll(List<> destination, List<> source)

Add the appropriate generic syntax to the signature. The method should work with any List<>.

### Examples

