## Runnable

Create a runnable implementation that prints 10.000 random lines in the provided file. Run the runnable in the main thread.

## File generator runnable

Create 10 files (file0.txt, ... file9.txt) each containing 10.000 random lines and run in the main thread.

## Files analyzer runnable

Create an anonymous runnable that analyze the provided files and prints which files contain which item from the provided strings. Run the runnable in the main thread

e.g
"random 1", "random 2", "random 3", "random 4"

```
file0 contains random 1
file0 contains random 2
file9 contains random 1
file9 contains random 3
```

## 10 files concurrent processing

Create a runnable concrete implementation that processes a single 10.000 random line containing file and prints how many lines contain "ab" substring in a given file.

Create 10 threads to process 10 files in parallel.

output example:

```
file0.txt 124
file7.txt 1000
file9.txt 2
```

## Shared List:

Create a runnable concrete implementation that processes a single 10.000 random line containing file and adds in a shared list string that is a concatenation of a file name and number of lines that contain "ab" substring in a given file.

Create 10 threads to process 10 files in parallel.

use join() method for each thread to wait for threads termination and print shared list content in the main thread

## Safe and Unsafe Car

Create Car interface which describes a car that has 6 passenger seats. The car should have the following methods:
add(String passengerName)
printPassangerNames()

Create an unsafe and a thread safe implementation of the Car interface.

## Shared list

Create 100 threads that accept anonymous runnable instances and each adds 1 to 500 numbers to a shared list.

Assure that after all threads execution list size is 50000.

## Custom safe list implementation

Create a SimpleList interface with following methods.
void add(T item);
int getSize();
T get(int index);

All implementations of the SimpleList interface should encapsulate an array of 10.000 objects.

Create SafeSimpleList and UnsafeSimpleList implementations.

Stress both implementations by 20 threads

## Binary file

Create an application that prints 2 random integers in a file in binary format and closes it. Move the main thread to sleep state for 10 seconds then read 2 integers from the file and append the sum of 2 integers to a file.

## Safe cartesian system

.

Create a safe CartesianSystem with methods to set coordinates and retrieve them so if we set (1,1) and (-1, -1), we will never be able to fetch (1, -1).

## List and Set size assertion

Create User classes with age, name classes.

Create 5 (1, "user1"), (2, "user2"), (3, "user3"), (2, "user2"), (2, "user2") users.

Create ArrayList, LinkedList, HashSet and TreeSet of the users mentioned above.

Write assertion method and assert that size of first 2 is 5 and size of hashSet and treeSet is 3.

## Numbers generator and printer

Create a method to return a list of sequential integers for the provided count and print 2xValue of each element using a separate list argument having method which can print any kind of numbers (Number is an abstract class in java.lang package)

Also, create a method to return a list of random doubles and print them using the method mentioned above.

```
input: 5, integer
output:
2.0
4.0
6.0
8.0
10.0
```

```
input: 2, double
output:
44.22
-8884.68
```

## User repository

Create a UserRepository interface and JDBCUserRepository implementation.

UserRepository has the following methods:

void save(User user)
void deleteAll()

## List and Set assertion

Срок сдачи: Вчера

Create a single method that checks the size of the provided data structure (should work at least for List and Set implementations such as ArrayList, LinkedList, HashSet, TreeSet) and throws an exception in case of size mismatch. The method must receive size as an argument.

## Book Proxy

Write ProxyBook class that extends Book(Book That we wrote in lesson) which must have
1.Boolean Flag and isLoaded method that return the flag.
2.Void method Load which change flag from subtask 1 to true value(if book is already loaded you need to throw custom exception )
3.Override methods getTitle() and getAuthor()  they must return the values if book is loaded otherwise they must throw