TESTABILITY ASSIGNMENT #3

Exercise 1

1. Write a program where you illustrate what non-testable code could look like. Make sure to include some code smells that you can refactor away later.

2. Static analysis:
   Use a tool to automatically check for code smells, e.g. CheckStyle and FindBugs for Java. You can use JaCoCo to calculate Cyclomatic Complexity. Document the result.

3. Rewrite the program with testability in mind and write automated unit tests for the program.

   In the transformation of the code structure, you can get inspiration from Martin Fowler's refactory pattern catalogue[1], e.g.

   https://refactoring.com/catalog/extractFunction.html
   https://refactoring.com/catalog/removeDeadCode.html
   https://refactoring.com/catalog/inlineVariable.html
   https://refactoring.com/catalog/introduceParameterObject.html
   https://refactoring.com/catalog/replaceConstructorWithFactoryFunction.html
   https://refactoring.com/catalog/replaceExceptionWithPrecheck.html

4. Static analysis - again:
   Use the same tool(s) as in 2) to automatically check your code again. Document the result.

[1] Also good to listen to the influencers – Martin Fowler is at top of list: https://apiumhub.com/tech-blog-barcelona/software-development-influencers/