The purpose of this assignment is to practice the development of Java programs with nested classes and interfaces.

Exercise 1 – Population

Import the *es1* folder into your IDE. The program creates a random list of people (Students, Workers and Person) and outputs statistics about the population. To compute the statistics, helper classes and interfaces from the *operations* package are used.
Your goal is to hide the implementation of those classes by removing the *operations* package and refactor all its interfaces and classes as nested classes / interfaces.
Please implement the solution in three different versions:

1. With nested static classes and nested interfaces inside the *S3Es1* class
2. With nested local classes in the *main* method
3. With anonymous classes in the *main* method

Note: Classes in the *person* package should remain untouched and focus should be set on removing the *operations* package only.

Exercise 2 – Advanced stack

Import the *es2/AdvancedStack* interface into your IDE. The interface declares, in addition to the common stack methods (*push*, *pop*, *peek*, *size* and *isEmpty*), a *count* method that accepts as parameter any object that implements the *AdvancedStack.Evaluate* interface. The method tests each element in the stack and counts how many elements in the stack are evaluated to true by the provided Evaluate object.

Write the *CustomStack* class that implements the *AdvancedStack* interface without using arrays or Collections (such as ArrayList, LinkedList, etc.). Instead, please introduce in the *CustomStack* class an own *StackElement* class as a list node to perform the storage of the pushed values. Keep in mind that the *StackElement* class has never to be visible/accessible by the client code of the *CustomStack* class, as its purpose is only internal to the stack's implementation logic.

To test your implementation, push a few objects (Strings, Integers, Float, ...) to the stack and verify that the *count* method returns the correct number of items by creating at least one anonymous class and one inner class implementation of the *AdvancedStack.Evaluate* interface.

Note: some possible *AdvancedStack.Evaluate* specialization classes could test:

- That the object is an instance of the Integer class
- That the object is a String containing "test"
- That the object is a Float or Double and its value is greater than 0.5
- That the object represents an even number