Programming Paradigms
Lab 10

Jacek Wasilewski

You are supposed to analyse, design and implement the following problems.

Generics

1. Create a class to store pairs of integers. Call it `IntPair`. It should take two parameters and store them.
2. You would like to have pairs of other types of elements as well, e.g. of String and Double. Create a generalised version of the `IntPair` class. Provide example of use.
3. You would like to compare elements that you are storing. Implement `isFirstGreater`, `isSecondGreater` and `areEqual` methods. Show on examples that these work on Int, Double and String types.
4. Check if your implementation supports Java classes, e.g. `java.land.Integer`. If not, fix your code.
5. Is your class using non-variant subtyping or co-variant subtyping? Create a variable that expects a pair of `scala.math.ScalaNumber` but create an instance of pair with `scala.math.BigInt`. Is it possible?
6. Modify your class to use co-variant subtyping.
7. Your `Pair` class takes two parameters of the same type. Create a parent class where you can specify type of each element separately.
8. Modify your current class to extend the one that you have just created. Note that you will not be able to move your methods to the parent class. Try to explain why.

Stateless and Stateful

1. We do shopping. We want to keep track of our shopping cart. We can add a product (product is just represented by its name) to our cart. We can also remove a product if we decide we do not need it. It is also possible to list all products that we have in our cart (just print them out), we can count how many products we have. It is possible to calculate how worthy is the cart - to make it simple, every single product worths 2 euro.

   Think if it makes sense to model this shopping cart as a stateful object or rather as a stateless object, then implement it and provide examples of use.

2. We have a simple calculator. We can add, subtract, multiply and divide numbers. Also we can calculate a square root and put a number to the power of 2. We can initialise our calculator with any number, then we perform an operation on this number. Next operations depend on previous results. We do not store the history and we do not provide the "memory" function.

   Think if it makes sense to model this calculator as a stateful object or rather as a stateless object, then implement it and provide examples of use.