Object-Oriented Programming
May 2019

Notes or documents of any kind forbidden. Duration: 3 1/2h. Please answer the questions on separate sheets labeled with your name, section, and student ID.

---

1. The problem consists in programming in Java a class `IntSequence` suited for representing a finite sequence of integer numbers. In other words, an instance of this class is characterized by a length $k \geq 0$ and by k integer numbers v_1, v_2, $\ldots v_k$. The order in which these numbers appear is relevant; for instance, the sequences $[1, 2, 3]$ and $[3, 2, 1]$ are considered to be different.

   The class `IntSequence` should satisfy the following requirements:

   - It must be possible to instantiate an empty sequence, a sequence containing a single given number, and a sequence corresponding to a given array of numbers.
   - It must be possible to append a sequence to another one, i.e., to turn the sequence $[v_1, v_2, \ldots, v_k]$ into $[v_1, v_2, \ldots, v_k, v'_1, v'_2, \ldots, v'_\ell]$ given another sequence $[v'_1, v'_2, \ldots, v'_\ell]$.
   - It must be possible to extract from a sequence a value at a given position, i.e., to obtain v_i from the sequence $[v_1, v_2, \ldots, v_k]$, given i such that $1 \leq i \leq k$.
   - It must be possible to extract from a sequence a subsequence of consecutive elements starting and ending at arbitrary positions, i.e., to construct the sequence $[v_i, v_{i+1}, \ldots, v_j]$ from the sequence $[v_1, v_2, \ldots, v_k]$, given i and j such that $1 \leq i \leq j \leq k$.
   - Instances of this class must be clonable, comparable to each other, and serializable. It must be possible to manipulate them simultaneously from separate threads.
   - In case of any error, a dedicated exception should be thrown.

   Note: You are free to implement any additional class required by your solution.

2. (All answers should be thoroughly justified.)

   (a) What are constructors? How do they differ from methods? In Java, can a constructor be invoked from another one, and if yes, under which conditions?

   (b) What is an abstract class? Can an abstract class define non-abstract methods?

   (c) A programmer defines a subclass `RealSequence` of the class `IntSequence` considered in Problem 1, in which the sequence now contains real instead of integer numbers. What is the particular application of inheritance used in this case? Is the substitution principle satisfied by this application?

   (d) In Java, what are checked exceptions? For what reason are they distinguished from runtime exceptions?

   (e) In Java, why is it forbidden to use the `instanceof` operator with a generic type?

   (f) Explain the principles of operation of the Java instruction

```
synchronized(v)
{
  ...
}
```

   where `v` is a reference to some object. What are the advantages of using this instruction over defining a method with the `synchronized` attribute?