Object-Oriented Programming
June 2017

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.

Question 1

A *polynomial* f of degree n with integer coefficients is a function

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0,$$

where, for each $i \in \{0, 1, \ldots, n\}$, $a_i \in \mathbb{Z}$ is the *coefficient* of the term of degree i, such that $a_n \neq 0$ if $n > 0$.

The problem consists in developing a Java library for representing and manipulating polynomials with integer coefficients. This library must contain at least a class `Polynomial`, satisfying the following requirements:

- An instance of this class represents a polynomial. It must not be possible to modify this polynomial after it has been created.
- It should be possible to create a new polynomial by providing either:
  - a single degree n and an integer coefficient a_n. The represented polynomial is then $a_n x^n$.
  - an array $[a_0\, a_1\, \ldots\, a_{n-1}\, a_n]$ of integer coefficients. The represented polynomial is then

    $$a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0,$$

    or
  - references to two polynomials $f_1(x)$ and $f_2(x)$. The represented polynomial then becomes equal to their sum $f_1(x) + f_2(x)$.
- One can extract the degree and coefficients of a polynomial.
- One can check whether a polynomial is identically equal to 0 (i.e., whether it contains only one term $a_n x^n$ such that $n = 0$ and $a_n = 0$).
- One can check whether two polynomials are equal (i.e., whether they have the same degree and coefficients).
- It should be possible to compute the value of a polynomial $f(x)$ for a given value of x.
- Instances of this class should be clonable.
- In the case of any error, a dedicated exception should be thrown.

Question 2

(All answers should be thoroughly justified.)

1. In Java, is it always possible to replace an abstract class by an interface?
2. When a Java programmer defines a new exception class, how does the compiler determine whether it belongs to the *checked exception* or *runtime exception* category?
3. How does Java determine which constructors are executed during the instantiation of a class? In which order are they executed?
4. Explain why the following Java code fragment is invalid:

```
public class MyList<T>
{
  public MyList()
  {
     T element = new T();
     ...
  }
  ...
}
```

5. Consider the following Java program :

```
public class Concurrent extends Thread
{
  private Object o1, o2;

  public Concurrent(Object o1, Object o2)
  {
    this.o1 = o1;
    this.o2 = o2;
  }

  public void run()
  {
    while (true)
    {
      synchronized (o1)
      {
        synchronized (o2)
          {
            ...
          }
      }
    }
  }
```

```
  public static void main(String[] args) throws Throwable
  {
    Integer i1 = new Integer(0);
    Integer i2 = new Integer(0);

    new Concurrent(i1, i2).start();
    new Concurrent(i2, i1).start();
  }
}
```

Is it possible for the execution of this program to end up in a deadlock? If yes, describe a corresponding execution scenario.