# Voortgezet Programmeren (FEB23007)

Example exam

Block 3 / Spring 2012

## Instructions

All the questions are *essay questions*: write concisely everything you know about the topic. Describe what the topic entails, and reflect on the impact of theory on the practice of programming. If you do not know something, it is better to write less than to try and invent an irrelevant answer. All questions should be answerable within 1 page of text with normal size hand writing (note, however, that you can write more as well).

In the actual exam, no extra material (neither dictionaries nor calculators) is allowed. Take with you only a pencil, a sharpener and an eraser.

NOTE: the real exam will contain 3 or 4 questions.

## Questions

1. Errors and exceptions in Java
2. Interfaces

## Example answer (q1)

Integral part of any programming language is the signalling of error conditions. This is especially important for method calls, as some can fail due to incorrect input or factors independent of the program(mer) (e.g. hard disk breaking down during file read). Traditionally in procedural programming the erronous method calls return a value outside the set of valid values. However, sometimes there are no such values as the set of valid values might be equal to all possible values for the return type (e.g. adding two integers together: the result is an integer and there are no integer values that are not valid for the summation result). Also, the languages cannot enforce the caller to explicitly check or even to store the return value, causing the mechanism to lead to error-prone code.

Although invalid return values can be used to signal error in method execution also in Java, the language also contains exceptions as an abstraction of the execution error. There are two types of exceptions: checked and unchecked. Checked ones have to be caught and handled or declared by the method to be thrown to its caller. They are used to signal error conditions from which the program must be able to recover (e.g. file not found while trying to read one). Unchecked exceptions usually signal more severe errors that the program shouldn't even try to recover from, such as violating pre-conditions.

## Example answer (q2)

Interfaces are an essential concept in object-oriented programming for enabling code re-use. The term interface can refer to the interface of a class (public methods), its inheritance interface (protected methods), or to "real" interface classes. The last ones are composed of method declarations that must be implemented by classes implementing the interface. Interface classes enable polymorphism as classes obtain, in addition to their own object type, also the object type of the interface, and can act as the interface type whenever needed. For example, a sorting algorithm does not need to know anything of the sorted objects except how they need to be ordered; this could be asked from the objects themselves (Comparable interface) or a sorting key could be asked from them (some type of Keyable interface with the method int getKey()). Then the algorithm could sort an array of objects implementing the interface using methods defined in the interface to define the objects' mutual order. Now in order to sort objects of a new class, it would need to just implement the interface and define the sorting order through the interface methods - the actual sorting algorithm continues working as it is without any modifications.