Institut for Matematik og Datalogi
Syddansk Universitet

November 14, 2011

Peter Schneider-Kamp

Programmering B
1st Weekly Note (E11, Week 45)

Format

The course is taught by Peter Schneider-Kamp. Lectures will be on Monday mornings 12:15–14 and on some Wednesdays shortly after sunrise 10:15–12.

There are four sections of students: matematik (S2), matematik-økonomi (M1), and datalogi (S7, S17). These sections will meet for both discussion sections and lab exercises taught by Philip Peters, Abyayananda Maiti, and Felix Palludan Hargreaves, respectively.

For a precise schedule, please see the web site of the course. As we will sometimes mix or swap discussion sections and lab exercises, please always carefully read the weekly notes such that you will end up in the right room. Note in particular that lab exercises will usually be held in IMADA's "Terminalrum" and not in the officially assigned room.

The weekly notes and other information about the course are available from the course home page:

`http://www.imada.sdu.dk/~petersk/DM503/`

You can also access the course home page through the university-wide e-learning system "Blackboard":

`http://e-learn.sdu.dk/`

Textbook
David J. Eck: *Introduction to Programming Using Java.* Version 6.0, Lulu, 2011. It is available as PDF and HTML:

`http://math.hws.edu/javanotes/`

Supplementary Reading
Allen B. Downey: *Think Java: How to Think Like a Computer Scientist.* Version 5.0.4, Green Tea Press, 2011. It is available as PDF and HTML:

`http://greenteapress.com/thinkapjava/`

Evaluation
Your progress in the material of the course is evaluated by a practical project. This project will consist of two parts. The first part will be probably handed out in week 46. Both parts will require you to model a problem, implement a program that solves the problem, and test your implementation.

Reading for Week 45

Chapter 1 and Sections 2.1–2.3.2, 2.4.6, 2.5.1–2.5.5, 2.5.8, 2.6, 3.1, 3.5, 3.6, 4.2.1–4.2.3, 4.3.1–4.3.2, 4.3.4, 4.4 of "Introduction to Programming Using Java"

Lecture: Monday, November 7, 12-14 (U20)

After an introduction to the course we will learn how to express the basic constructs of imperative programming languages in Java. I.e., we will learn about variables, assignment, serial execution, conditional execution, loops, and function declarations. The concept of static type declarations will be introduced.

Discussion: see detailed schedule on course home page

Meet in the assigned room. For M1 and S2, the teaching assistants will introduce themselves.

In all sections, you should start by a comparison of the basic imperative constructs between the Python and the Java language. Discuss the similarities and the differences. Discuss the following three questions: What is the difference between "dynamic" and "static" types? What are the advantages of "dynamic" and "static" types? What behavior is enforced by static types?

Take the Chapter 1 quiz from the course book and discuss questions that you are unsure about.

Then move to IMADA's terminalrum and write your first Java program: `HelloWorld.java`

```
/**
 * The Java version of Python's print "Hello World!"
 */
public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

The main work is done by one line, that is by:

```
System.out.println("Hello World!");
```

This line write the string `"Hello World!"` to the screen.

Compilation and Execution of Java programs

A compiler is a program that translates a text representation of a program written in a programming language into a more machine-readable format. While this step is implicitly transformed by Python's interpreter `python`, for most language this has to be done, before the program can be executed. Files that contain Java code always *have* to end with `.java`, as otherwise the compiler will refuse to work.

To compile the Java program described in the text file `HelloWorld.java` into the machine-readable program`HelloWorld.class`, use the command `javac HelloWorld.java`. To execute the compiled program, use the command `java HelloWorld` that executes the file `HelloWorld.class`. Note that the ending `.class` has to be omited.

Familiarize yourself with the Java documentation:

```
http://download.oracle.com/javase/6/docs/
```

The most important part of the documentation is the API documentation. Find the documentation for the class `Scanner`. Read the examples at the top and have a look at which kind of elements can be read by the different `nextXXX()` methods.

Modifiy `HelloWorld.java` using the topmost example from the `Scanner` documentation and the method `nextLine()` to prompt the user for a name and then greet the user by saying `Hello Mette!` if the user entered `Mette`.

If you are finished with this, continue with Exercise 2.3 from Chapter 2 of the course book.

Lab: see detailed schedule on course home page

Solve Exercises 2.1, 2.2, 2.4, and 2.5 from the course book and take the quiz for Chapter 2 for all questions that you were supposed to read up on. Then solve Exercises 3.1–3.4. In all cases where `TextIO` is used, use the `Scanner` class instead.

Then solve as many of the following exercises from the supplementary reading "Think Java" as possible: Exercise 3 of Chapter 1, Exercises 1–3 of Chapter 2, Exercises 2–4 of Chapter 3, and Exercises 3–5 of Chapter 4.