Introduction to Programming

Week 15

Magnus Madsen

Week 15: Outline

Monday

- Modern Java: Lambdas, Streams, Records, Pattern Matching, Type Inference

Thursday

- Wrap-Up and Exam

Quote of the Week

"That language is an instrument of human reason, and not merely a medium for the expression of thought, is a truth generally admitted."

— George Boole

Epigram of the Week

"If a listener nods his head when you're explaining your program, wake him up."

— Alan Perlis

Modernizing Java with Functional Programming

TODO 🐑 ML: Java is old, but was pushed to modernize because of Kotlin, Scala

Lambdas and Functional Interfaces

TODO 🐑 **ML:** already saw a bit about Lambdas last week

TODO 🐑 **ML:** what is a lambda, really?

Streams

TODO 🐑 **ML:** examples, compare with loops

TODO 🐑 **ML:** note Pureness

TODO 🐑 **ML:** things go wrong without pureness

Records

TODO 🐑 **ML:** compare with boilerplate stuff

Pattern Matching

TODO 🐑 ML:

Type Inference

TODO 🐑 ML: var

TODO 🐑 ML: show where it cannot infer

Course Description

Course content

Foundational: Understanding what a Java program is, how to compile it, and how to execute it. Reasoning about whether a program is well-formed and about its behavior.

Imperative Concepts: Writing simple methods using local variables, if-then-else, and for- and while loops. Writing simple data structures using plain objects and arrays.

Object-Oriented Concepts: Writing simple classes with encapsulated state, getters and setters, and using interfaces and inheritance.

Programming Techniques: Programming with common data structures, such as lists, sets, and maps. Applying basic debugging and testing techniques to understand how a program behaves. Manipulating the file system, including the creation, reading, and writing of files.

Learning objectives (1/2)

After the course, students will be able to:

- Explain how to write a Java computer program, compile it, and execute it.
- Use elementary imperative programming language constructs, including: primitive data types, local variables, assignment, arrays, if-then-else, and for- and while loops.
- Use elementary object-oriented programming language constructs, including: classes, interfaces, objects, and methods.
- Use common data structures such as lists, sets, and maps.

Learning objectives (2/2)

After the course, students will be able to:

- Identify, explain, and overcome compiler errors (e.g. syntax, semantic, or type errors).
- Apply programming techniques to write small programs in imperative or object-oriented style.
- Apply basic debugging and testing techniques to understand and correct program behavior.
- Apply advanced programming features such as inheritance and generics.

Exam

Exam

The exam consists of two parts:

> **i** **Part A:** An individual take-home programming project. The exam is open-book, i.e. students may use all materials available except Generative AI. Students may discuss the project with each other, but may not share any source code.

> **i** **Part B:** An individual written exam. The exam is closed-book, i.e. students may not use any materials. The scope of the written exam is the entire course syllabus plus the take-home programming project.

The two parts take place at different times and places.

- You have three days for the take-home programming project.
- You have two hours for the written exam.

Exam

You obtain **one grade** weighted approximately equally between **Part A** and **Part B**.

The Danish Grading System. The so-called “7-trins skalaen”:

Grade	Danish	English	ECTS
12	Fremragende	Outstanding	A
10	Fortrinligt	Excellent	B
7	Godt	Good	C
4	Jævnt	OK	D
02	Tilstrækkeligt	Adequate	E
00	Utilstrækkeligt	Inadequate	Fx
−3	Ringe	Poor	F

cf. https://ufm.dk/en/education/the-danish-education-system/grading-system

How to prepare for the exam

✓ I have read and understood the textbook and other material.

✓ I have solved all weekly exercises and understood them.

✓ I have completed all mandatory hand-ins and understood the feedback from the TA.

✓ I have attended lectures, TA classes, and the study cafe.

✓ Whenever I did not understand something, I asked for help.

If you do these things, you will be successful 🌈 🎉 🥳.

Exam

Q: Write seven Java keywords.

Q: What is double buffering?