Course Wrap-up

Learning a new language

A new memory model.

Static typing rather than dynamic typing.

Compiled rather than interpreted.

Primitives vs. only objects.

Everything belongs to a class.

Generics.

Interfaces and abstract classes.

You also did more of the learning yourself than before.

More languages

You'll see more languages and the underlying language design principles and approaches:

- CSC209: Software Tools and Systems Programming
- CSC324: Principles of Programming Languages
- CSC343: Introduction to Databases
- CSC309: Programming on the Web
- CSC488: Compilers and Interpreters

Value:

- Knowing which tool to apply in a given situation
- Deeper appreciation of a language's design and how best to use it

Tools and Technologies

New software development tools:

- A fully-featured IDE
- Version control

Design Patterns

- Language-independent
- Generic solutions to recurring design problems
- Solutions that follow the object-oriented design principles and best practices
- More design patterns in CSC301: Introduction to Software Engineering

Regular expressions

These have many practical applications:

- Parsing input in a Java program (or another language).
- Many Unix commands, such as grep, use regular expressions.
- Regular expressions and related concepts are important in language design and compilers. CSC488: Compilers and Interpreters

Learn more about the underlying theory in CSC236: Introduction to the Theory of Computation and CSC448: Formal Languages and Automata.

Floating point issues

Floating point calculations are inherently inexact

- + is not associative!

CSC336: Numerical Analysis

SOLID

SOLID: five basic principles of object-oriented (Developed by Robert C. Martin, affectionately known as “Uncle Bob”.)

- Single responsibility principle
- Open/closed principle
- Liskov substitution principle
- Interface segregation principle
- Dependency inversion principle

More software design

You will learn much more through:

- Internships
- Open source contributions
- CSC301: Introduction to Software Engineering
- CSC302: Engineering Large Software Systems
- Project courses (CSC494/495 or CSC490)
- PEY

Exam prep

- 2 hours
- Closed book
- Javadoc and internal comments not required (except where mentioned)
- Import statements not required (except where mentioned)
- Helper methods always welcome
- See the practice exam!

Ideas for how to prepare

- Solve old tests and exams
- Revisit the assignments and projects and learn from what you did previously; improve it
- Write code that tries to poke holes in your understanding of Java concepts (name lookup, exceptions, etc.)
- Write regular expressions and visit regexcrossword.com
- Write code that uses the design patterns we studied
- Write code on paper!

Course Evaluations

Very important! They’re used by:

- Future students in choosing courses
- Faculty for improving the course
- University for evaluating faculty

Please complete them!

Thanks for a great semester!