Software Engineering

Revision

Outline

- Module overview
- Part I
- Part II
- Final exam

Module overview

- A *practical introduction* to **software engineering**
- Consists of two parts:
  - Part I: intermediate-advanced OOP
  - Part II: introduction to software engineering

Part I concepts & techniques

- More abstractions:
  - type: generalisation (super-type/sub-type)
  - iteration
  - polymorphic
- Exception
- Unit testing and debugging

Part I (1)

- Abstractions:
  - *type*: extends, implements
  - *iteration abstraction*: `java.util.Iterator`
  - *polymorphic abstraction*: polymorphic procedure and polymorphic type (e.g. Set)
- Exception:
  - create exception: `Exception` or `RuntimeException`
  - design with exception: throw/handle

Part II

- Theory
- Practice
- Project

Software engineering theory

- SE method: structured and well defined process
- Key techniques learnt in each phase:
  - requirement engineering: analysis & specification
  - object oriented design with UML
    - <u>iterative</u> “decomposition by abstractions”
  - design review & implementation plan
  - incremental testing with JUnit 4

Software engineering practice

- Case study: Keyword search engine (KEngine)
- Program examples:
  - Xref
  - SpellChecker
  - PathFinder

Software project

- Goal: to develop
  - an object-based keyword search engine
  - using the keyword search engine as a library
- Approach:
  - *incremental*: complete each stage as the module progresses
  - finish the software in the final week!
- Deliverables: a software product with technical report (design note)

Final exam

- Time: 90 - 120 mins (TBC)
- Format: closed book, paper-based
- Consists of two parts:
  - Part I: practice tasks
  - Part II: multiple choice
- Practice tasks:
  - design, implementation of OOPs
  - may be related to KEngine
- Part II: random questions about various topics

Q & A