Large Scale Programming
1DL410, autumn 2009

Cons T Åhs (cons.t.ahs@it.uu.se)
TA: Manivasakan Sabesan (sabesan@it.uu.se)

http://www.it.uu.se/edu/course/homepage/storprog/ht09
Presentations

- Lecturer: Cons T Åhs (more later)
- TA: Manivasakan Sabesan
Cons T Åhs

- MSc Engineering Physics, computing science
- Former lecturer at Uppsala University, (DVL/P, ITP..)
  - foundations, algorithms, functions, relations, objects, compilers, theories, proofs, pragmatics, ..
- Private contractor/consultant with experience from
  - music notation
  - speech synthesis
  - 3D graphics
  - medical imaging
  - financial systems
  - unix network programming
  - real time video decoding
  - compiler writing, programming language implementation
  - teaching outside the university (Ericsson, SUN, ..)
- Currently working in the gaming industry with development and responsibility for educating developers in advanced design and programming.
What’s this course about?

The official course description:

To write a small program for yourself or as an assignment is one thing.

It is another thing to develop a larger program of high quality with a team.

In this course you won't learn how to program - you should know that - but how to program well.

You learn how to write programs with few errors that can be maintained.

Most of the course is a project, constructing a larger program in a team.

During this process, your program will be inspected and discussed several times.

You will receive constructive criticism so that you can improve the program.

Programming is more than coding: testing, version management, profiling and optimizing, and documentation are part of the course.
What’s this course about?

Writing great and beautiful programs!

- Understandable - concise and elegant code, documented
- Tested and Testable - automated tests brings confidence and trust to the code
- Maintainable - the program should not fight back when you need to change it or fix bugs (both of these will happen)
- Safe - behaves reasonably when things start to fall apart, i.e., does not fall apart itself.
- Correct - does what it is supposed to do and nothing else
- Efficient - acceptable use of resources, i.e., time and space
- Reusable - it should be possible to easily reuse parts of the program in another project
- Portable - it should be possible to run the program on another CPU, OS, etc without problems (this essentially comes for free as we’re using Java)
Beautiful programs - how?

- Use methods, conceptual tools
  - abstraction
  - OO principles, design patterns..
- Use tools
  - IDE, version control, profiler, debugger, scripts etc
- Automation
  - tests, building, ..
- “Correct” mindset - Att tänka fritt är stort, att tänka rätt är större
  - change will happen
  - you will have bugs
  - it is not as simple as you think
  - you will forget the details
  - your code will wither and rot
  - you will have to bad ideas that you need to retract
What to expect and not

- **Expect**
  - conceptual tools
  - methods
  - principles
  - references to tools
  - questions, challenges

- **Do not expect**
  - large code examples
  - detailed lessons on using tools

- **What you must do**
  - spend time learning the tools (Eclipse, ant, svn, XML, ..)
  - use the tools efficiently
Assignment

- Groups of three (3) - no exceptions. Form groups now and notify the TA of the group members.
- Firm deadlines - again, no exceptions.
- Write a ray tracer
- Five parts, which different motivations
  1. Abstraction, beautiful programs
  2. Basic ray tracer, XML, using tools
  3. More complete ray tracer (reflection, shadowing), extending an existing program
  4. optimisation, refactoring
  5. make a movie, parallelise ray tracer
Grading of assignment

- Fail (U) - you flunk the course. This only happens if you don’t take the work seriously or lack basic qualifications.
- Revise (K) - the work has shortcomings that need to be correct at once. Resumbmit a new solution as soon as possible.
- Pass and revise (GK) - the work has minor shortcomings that need to be correct when the next part is handed in.
- Pass (G) - good work, but there might still be useful comments.
Assessment

- To pass the course (grade 3):
  - pass the project, i.e., all parts of the assignment
  - pass the home exam (with questions about your particular project)
- To pass the course with distinction (grade 4):
  - pass the project and the home exam with at least grade 4
- To pass the course with even more distinction (grade 5):
  - pass the assignment, home exam and a traditional exam (with more theoretical question)
  - the grade is the average of the project and the two exams.
Why a ray tracer?

- Complete chain from input (XML) to output (an image)
- Apply and combine knowledge from previous courses.
- Tangible result - you can mail your images to your parents.
- The problem domain is simple and the basic algorithm is very easy to describe.
- Computationally interesting and intense - the basic algorithm is rather easy to implement, but it can be optimised further in many ways.
- Correctly done, you will have a well crafted program that can be extended with, e.g., new objects without any changes to code.
- No GUI, which might be considered a drawback.