COMP3111 -- Introduction to Software Engineering

Learn. Pratice. Have fun!

What's COMP3111

- Introduction level
  - Developing large software systems by many people in a long time;
  - Basic thoery, model, practice
  - Practice using real world languages, tools, and methods
- Difference between software engineering and programming

What we going to cover (Thoery)

- How to develop large software as a team effort
  - Non-technical aspects
    - As a process of coordinating human activities (macroscopic)
    - As a result of effective communictations (socio-technical)
      - With client
      - With team members
  - Technical aspects
    - As a set of drawings (modeling)
    - As a collection of clever design ideas (patterns)
    - As a correct instructions to computers (testing)

COMP3111 Syllabus (Theory)

- Introduction to software engineering
- Software development process
- UML modeling
- Requirements capture
- Design patterns
- Testing

Syllabus (Pratice)

- Basic Java programming
- Java collections framework
- Java GUI programming with Swing
- CVS environment in Eclipse
- Java I/O
- Java Unit testing
- Finding bugs in your code

How do you learn both?

- Lecture
  - 1 hour + 20 mins for special needs
- Tutorial
  - Review practice materials in lecture
  - More detailed and advanced topics
- Lab
  - Solve the posted problem, not graded
  - Practice with the help of TA
- Project
  - Use the technique in your project
- Course etiquette
  - Lecture: if you really want to talk, talk to me.
  - Project: independent work or steal smartly

Grading

Item	Due Date	Weight
Midterm	Mar 17	20%
Project	Part I (Apr 3) Part II (May 8)	40-48%
Final	TBD	40%
Bonus	Will explain	4%

You guessed right! Attending lecture/tutorial/lab is optional!

The Project

- Group work
  - Group of 4. You can form your own group. Do it before Feb. 17 or I will do it for you
  - Sign up sheet posted on Google doc, link on course website
  - Each team will be assigned a TA as mentor
- Objective
  - A fully functional multi-user Calendar
  - Completion in two major steps
  - Evaluated by acceptance test cases and documentation

Project Organization

- Given the source code of the Calendar GUI and a detailed HIGH LEVEL specification
- Complete the project in two phases:
- Phase 1: Single-user memory based scheduling
  - Learn to program in modern object oriented language such as Java using a fully featured integrated development environment (IDE).
  - Learn to convert imprecise textual technical requirements to precise technical specifications.
  - Learn to extend the given code base by adding new GUI elements and major new features.
  - Learn to develop software as a controlled and monitored process through the use of version control.

Project Organization (Cont.)

- Phase 2: Multi-user disk-based scheduling
  - Learn to implement a sophisticated set of functionalities.
  - Learn to support the notion of persistence and data management.
  - Learn to use unit testing and regression testing to maintain the quality of the software.
  - Learn to articulate and document the design and architecture of software systems.

Bonus Competition

- You can earn bonus for the project by implementing a feature/functionality not yet specified
  - Must be a new way of interacting with the user
  - For example, an Android version of the calendar is not counted as bonus
- Top 6 teams will get the bonus
  - Inform your mentor any time in the term to enter the competition
  - Submit your work at the end of the Phase II
  - The teaching team will sit together and rank all participants
  - Top 3 teams will get the full bonus, the next 3 will get 50% of the full bonus