Testing and Debugging
Project 2:
Mutation Testing
Mutation Testing

- Project will start next week.
- Presentation today because of holiday.
- Project will be available from SVN next week thursday.
3 Phases

• Transform source files.
• Compile the transformed files.
• Run tests for all mutations.
Mutation Operators

- Relational Operator Replacement (ROR)

- Constant Replacement (CRP)
Assumption

- Use mutant schemata
- Only one mutation per operator/constant
- No higher order mutants required
- => Makes implementation easier
Example

if( (a > b) || ( x >= 5) )

if( (a < b) || ( x >= 5) )

if( (a > b) || ( x <= 5) )

if( (a > b) || ( x >= 0) )
Example

```c
if( (a > b) || ( x >= 5) )
    // Code block
```

```c
if( m1 ? (a < b) :(a > b) ||
    m2 ? (x <= 5) : m3 ? (x >= 0) : (x >= 5) )
    // Code block
```
Type of Variables

int i = 1;
double x = (i + 1.) / 2

int i = 1;
double x = (i + 0) / 2

int i = 1;
double x = (i + 0.) / 2

int i = 1;
double x = (i + 0.) / 2
Compiling

- Mutated source files are compiled by our test script.
- `ant check-mutation`
Executing Mutations

- Check if none of the tests fails.
- Run all tests on every mutant.
- Record the results.
Detecting Mutations

• Extend test cases from first project for Variance class such that it detects all detectable mutants.

• Mutation in serialVersionUID not expected to be detectable.
Project 1 - Fault localization

- You do not need to connectClasses to tests.
- Just take all coverage data that is in the specified directory.
Utility Methods

- `org.apache.commons.io.FileUtils`, provides utility methods to read write files.
- Collections can be converted to arrays using the `toArray()` method.
- `org.apache.commons.lang.ArrayUtils` provides utility methods to convert arrays, e.g. `Integer[]` to `int[]`. 
Static Initializer

- Called once when the class is loaded.
- Java Language Specification:

  8.7 Static Initializers

  Any static initializers declared in a class are executed when the class is initialized and, together with any field initializers (§8.3.2) for class variables, may be used to initialize the class variables of the class (§12.4).

```java
class A {
    static {
        // set up code
    }
}
```
Secret Tests

• Some constructs will not be in the secret tests:
  • synchronized blocks
  • assert statements
  • empty statement (;)
  • labels
  • class declarations inside methods (e.g. class SomeClass {});