Assignment 7

• Due Sunday, November 10, 2013
Software Qualities
Aspects of software quality

[Lewis & Chase]

- **Correctness**: The degree to which software adheres to its specific requirements.
- **Reliability**: The frequency and criticality of software failure.
- **Robustness**: The degree to which erroneous situations are handled gracefully.
- **Usability**: The ease with which users can learn and execute tasks within the software.
- **Maintainability**: The ease with which changes can be made to the software.
- **Reusability**: The ease with which software components can be reused in the development of other software system.
- **Portability**: The ease with which software components can be used in multiple computer environments.
- **Efficiency**: The degree to which the software fulfills its purpose without wasting resources.
Code Quality

Good

• shorter
• easier to read
• elegant
• clean
• modular

Bad

• longer
• harder to read
• less elegant
• tangled
• less modular (different levels of abstraction mixed up in code)
Reliability Testing
Computing better hash codes
hashCode from Object

http://docs.oracle.com/javase/6/docs/api/java/lang/Object.html

public int hashCode()

Returns a hash code value for the object. This method is supported for the benefit of hashtables such as those provided by java.util.Hashtable.

The general contract of hashCode is:

- Whenever it is invoked on the same object more than once during an execution of a Java application, the hashCode method must consistently return the same integer, provided no information used in equals comparisons on the object is modified. This integer need not remain consistent from one execution of an application to another execution of the same application.

- If two objects are equal according to the equals(Object) method, then calling the hashCode method on each of the two objects must produce the same integer result.

- It is not required that if two objects are unequal according to the equals(java.lang.Object) method, then calling the hashCode method on each of the two objects must produce distinct integer results. However, the programmer should be aware that producing distinct integer results for unequal objects may improve the performance of hashtables.

As much as is reasonably practical, the hashCode method defined by class Object does return distinct integers for distinct objects. (This is typically implemented by converting the internal address of the object into an integer, but this implementation technique is not required by the JavaTM programming language.)

Returns: a hash code value for this object.
Definitions

• If `f1.hashCode() == f2.hashCode()` but `f1` and `f2` are not equal, then the hash codes for `f1` and `f2` are said to collide.

• The collision probability is the conditional probability that `f1.hashCode() == f2.hashCode()` given that `f1` and `f2` are not equal.