Hypotenuse – Grading hints

This document is for instructors and grading assistants only.
Do not give this to students!

Background: This assignment is not a real assignment but demonstrates the use of various grading aspects with Graja.

The submission is graded with respect to several aspects:

Category	Aspect	Maximum score	Graded by module …
Functional correctness	A correct submission can calculate squares	35	JUnit
	A correct submission can calculate the length of a hypotenuse	25	
Maintain-ability	A correct submission declares private attributes only	15 for class Hypo and 15 for class SquareImpl	JUnit
	A correct submission uses delegation as a technique of reuse	40	JUnit with Mockito
	TODO	TODO	PMD
	Readability	15	Manually graded
	Visual Code layout	10	
Security	A correct submission declares private attributes only	10 for class Hypo and 10 for class SquareImpl	JUnit
	TODO	TODO	PMD
Reliability	A correct submission rejects illegal parameters	20 for cathetes >0 and 20 for Square parameter not null	JUnit
	A correct submission checks for double overflows	20 for square method	
	A correct submission has reasonable javadocs about valid parameter ranges	10	Manually graded
UML	A correct submission includes a correct UML diagram	40	Manually graded

Sample solution:

File `SquareImpl.java`:

```
import org.myinstitution.myassignments.hypotenusev02.Square;

/**
 * This class can square a double value and checks for overflows.
 */
```

```java
public class SquareImpl implements Square {
	private void checkOverflow(double v) {
		if (Math.abs(Math.sqrt(v*v) - Math.abs(v)) > 1E-5)
			throw new ArithmeticException();
	}
	/**
	 * This method calcluates the square of a given value.
	 * @param v value to be squared.
	 * @return v*v
	 * @exception ArithmeticException when v cannot be squared because of an
arithmetic overflow.
	 */
	@Override public double square(double v) {
		checkOverflow(v);
		return v*v;
	}
}
```

File Hypo.java:

```java
import org.myinstitution.myassignments.hypotenusev02.Square;

/**
 * This class can calculate the length of the hypotenuse of a right triangle.
 */
public class Hypo {
	/**
	 * Calculates the hypotenuse of a right triangle.
	 * @param s this object is needed for delegating square calculations. Must
not be null.
	 * @param a the first cathete. Must be > 0.
	 * @param b the second cathete. Must be > 0.
	 * @return the length of the hypotenuse.
	 * @exception IllegalArgumentException is thrown, when an illegal parameter
was detected.
	 */
	public static double hypo(Square s, double a, double b) {
		if (s == null) throw new IllegalArgumentException();
		if (a<=0 || b<=0) throw new IllegalArgumentException();
		return Math.sqrt(s.square(a) + s.square(b));
	}
}
```

Class diagram: