Ein weiteres Einreichungsbeispiel zum "JUnit-Backend" - LittleMath.java -

Christian Baumann

12. Januar 2008

Inhaltsverzeichnis

1 Fehlerlose Beispieleinreichung:

```java
public class LittleMath {
    /**
     * Returns absolute value of type long.
     * @param l Long input
     * @return Absolute value of l.
     */
    public long abs(long l){
        if(l < 0) return -l;
        return l;
    }

    /**
     * Returns absolute value of type double.
     * @param d Double input
     * @return Absolute Value of d.
     */
    public double abs(double d){
        if(d < 0) return -d;
        return d;
    }

    /**
     * Returns absolute value of type float.
     * @param f Float input
     * @return Absolute Value of f.
     */
    public float abs(float f){
        return (float)abs((double)f);
    }

    /**
     * Returns absolute value of type int.
     * @param i Int input
     * @return Absolute Value of i.
     */
    public int abs(int i){
        return (int)abs((long)i);
    }

    /**
     * Returns the maximum of two given long values.
     * @param l1 Long input.
     * @param l2 Long input.
     * @return Maximum of l1 and l2.
     */
    public long max(long l1, long l2){
        if (l1 < l2) return l2;
        return l1;
    }

    /**
     * Returns the maximum of two given int values.
     * @param i1 Int input.
     * @param i2 Int input.
     * @return Maximum of i1 and i2.
     */
    public int max(int i1, int i2){
        return (int)max((long)i1, (long)i2);
    }

    /**
     * Returns the maximum of two given double values.
     * @param d1 Double input.
     * @param d2 Double input.
     * @return Maximum of d1 and d2.
     */
    public double max(double d1, double d2){
        if (d1 < d2) return d2;
        return d1;
    }

    /**
     * Returns the maximum of two given float values.
     * @param f1 Float input.
     * @param f2 Float input.
     * @return Maximum of f1 and f2.
     */
    public double max(float f1, float f2){
        return (float)max((double)f1, (double)f2);
    }

    /**
     * Returns the minimum of two given long values.
     * @param l1 Long input.
     * @param l2 Long input.
     * @return Minimum of l1 and l2.
     */
    public long min(long l1, long l2){
        if (l1 < l2) return l1;
        return l2;
    }

    /**
     * Returns the minimum of two given int values.
     * @param i1 Int input.
     * @param i2 Int input.
     * @return Minimum of i1 and i2.
     */
    public int min(int i1, int i2){
        return (int)min((long)i1, (long)i2);
    }

    /**
     * Returns the minimum of two given double values.
     * @param d1 Double input.
     * @param d2 Double input.
     * @return Minimum of d1 and d2.
     */
    public double min(double d1, double d2){
        if (d1 < d2) return d1;
        return d2;
    }

    /**
     * Returns the minimum of two given float values.
     * @param f1 Float input.
     * @param f2 Float input.
     * @return Minimum of f1 and f2.
     */
    public float min(float f1, float f2){
        return (float)min((double)f1, (double)f2);
    }
}
```

2 Einreichung mit syntaktischem Fehler:

```
public class LittleMath {
    /**
     * Returns absolute value of type long.
     * @param l Long input
     * @return Absolute value of l.
     */
    public long abs(long l){
        if(l < 0 return -l;          //Error will occur!
        return l;
    }

    /**
     * Returns absolute value of type double.
     * @param d Double input
     * @return Absolute Value of d.
     */
    public double abs(double d){
        if(d < 0) return -d;
        return d;
    }

    /**
     * Returns absolute value of type float.
     * @param f Float input
     * @return Absolute Value of f.
     */
    public float abs(float f){
        return (float)abs((double)f);
    }

    /**
     * Returns absolute value of type int.
     * @param i Int input
     * @return Absolute Value of i.
     */
    public int abs(int i){
        return (int)abs((long)i);
    }

    /**
     * Returns the maximum of two given long values.
     * @param l1 Long input.
     * @param l2 Long input.
     * @return Maximum of l1 and l2.
     */
    public long max(long l1, long l2){
        if (l1 < l2) return l2;
        return l1;
    }

    /**
     * Returns the maximum of two given int values.
     * @param i1 Int input.
     * @param i2 Int input.
     * @return Maximum of i1 and i2.
     */
    public int max(int i1, int i2){
        return (int)max((long)i1, (long)i2);
    }

    /**
     * Returns the maximum of two given double values.
     * @param d1 Double input.
```

```java
     * @param d2 Double input.
     * @return Maximum of d1 and d2.
     */
    public double max(double d1, double d2){
        if (d1 < d2) return d2;
        return d1;
    }

    /**
     * Returns the maximum of two given float values.
     * @param f1 Float input.
     * @param f2 Float input.
     * @return Maximum of f1 and f2.
     */
    public double max(float f1, float f2){
        return (float)max((double)f1, (double)f2);
    }

    /**
     * Returns the minimum of two given long values.
     * @param l1 Long input.
     * @param l2 Long input.
     * @return Minimum of l1 and l2.
     */
    public long min(long l1, long l2){
        if (l1 < l2) return l1;
        return l2;
    }

    /**
     * Returns the minimum of two given int values.
     * @param i1 Int input.
     * @param i2 Int input.
     * @return Minimum of i1 and i2.
     */
    public int min(int i1, int i2){
        return (int)min((long)i1, (long)i2);
    }

    /**
     * Returns the minimum of two given double values.
     * @param d1 Double input.
     * @param d2 Double input.
     * @return Minimum of d1 and d2.
     */
    public double min(double d1, double d2){
        if (d1 < d2) return d1;
        return d2;
    }

    /**
     * Returns the minimum of two given float values.
     * @param f1 Float input.
     * @param f2 Float input.
     * @return Minimum of f1 and f2.
     */
    public float min(float f1, float f2){
        return (float)min((double)f1, (double)f2);
    }
}
```

3 Einreichung mit semantischem Fehler:

```java
public class LittleMath {
    /**
     * Returns absolute value of type long.
     * @param l Long input
     * @return Absolute value of l.
     */
    public long abs(long l){
        if(l < 0) return l;          //Semantically wrong
        return l;
    }

    /**
     * Returns absolute value of type double.
     * @param d Double input
     * @return Absolute Value of d.
     */
    public double abs(double d){
        if(d < 0) return -d;
        return d;
    }

    /**
     * Returns absolute value of type float.
     * @param f Float input
     * @return Absolute Value of f.
     */
    public float abs(float f){
        return (float)abs((double)f);
    }

    /**
     * Returns absolute value of type int.
     * @param i Int input
     * @return Absolute Value of i.
     */
    public int abs(int i){
        return (int)abs((long)i);
    }

    /**
     * Returns the maximum of two given long values.
     * @param l1 Long input.
     * @param l2 Long input.
     * @return Maximum of l1 and l2.
     */
    public long max(long l1, long l2){
        if (l1 < l2) return l2;
        return l1;
    }

    /**
     * Returns the maximum of two given int values.
     * @param i1 Int input.
     * @param i2 Int input.
     * @return Maximum of i1 and i2.
     */
    public int max(int i1, int i2){
        return (int)max((long)i1, (long)i2);
    }

    /**
     * Returns the maximum of two given double values.
     * @param d1 Double input.
     * @param d2 Double input.
     * @return Maximum of d1 and d2.
     */
    public double max(double d1, double d2){
        if (d1 < d2) return d2;
        return d1;
    }

    /**
     * Returns the maximum of two given float values.
     * @param f1 Float input.
     * @param f2 Float input.
     * @return Maximum of f1 and f2.
     */
    public double max(float f1, float f2){
        return (float)max((double)f1, (double)f2);
    }

    /**
     * Returns the minimum of two given long values.
     * @param l1 Long input.
     * @param l2 Long input.
     * @return Minimum of l1 and l2.
     */
    public long min(long l1, long l2){
        if (l1 < l2) return l1;
        return l2;
    }

    /**
     * Returns the minimum of two given int values.
     * @param i1 Int input.
     * @param i2 Int input.
     * @return Minimum of i1 and i2.
     */
    public int min(int i1, int i2){
        return (int)min((long)i1, (long)i2);
    }

    /**
     * Returns the minimum of two given double values.
     * @param d1 Double input.
     * @param d2 Double input.
     * @return Minimum of d1 and d2.
     */
    public double min(double d1, double d2){
        if (d1 < d2) return d1;
        return d2;
    }

    /**
     * Returns the minimum of two given float values.
     * @param f1 Float input.
     * @param f2 Float input.
     * @return Minimum of f1 and f2.
     */
    public float min(float f1, float f2){
        return (float)min((double)f1, (double)f2);
    }
}
```

4 Unit-Tests:

```
private ${CLASS} c;

@Before public void setUp(){
    c = new ${CLASS}();
}

@After public void tearDown(){
    c = null;
}

@Test public void longAbs(){
    assertEquals("abs(long) did not return the correct result.", 10l, c.abs(10l));
    assertEquals("abs(long) did not return the correct result.", 10l, c.abs(-10l));
}

@Test public void intAbs(){
    assertEquals("abs(int) did not return the correct result.", 99, c.abs(99));
    assertEquals("abs(int) did not return the correct result.", 99, c.abs(-99));
}

@Test public void doubleAbs(){
    assertEquals("abs(double) did not return the correct result.", 0.99d, c.abs(0.99d));
    assertEquals("abs(double) did not return the correct result.", 0.99d, c.abs(-0.99d));
}

@Test public void floatAbs(){
    assertEquals("abs(float) did not return the correct result.", 0.099f, c.abs(0.099f));
    assertEquals("abs(float) did not return the correct result.", 0.099f, c.abs(-0.099f));
}

@Test public void longMax(){
    long l1 = 9999l;
    long l2 = 1111l;

    assertEquals("max(long) did not return the correct result.", l1, c.max(l1, l2));
    assertEquals("max(long) did not return the correct result.", l2, c.max(-l1, l2));
}

@Test public void intMax(){
    int i1 = 234;
    int i2 = 123;

    assertEquals("max(int) did not return the correct result.", i1, c.max(i1, i2));
    assertEquals("max(int) did not return the correct result.", i2, c.max(-i1, i2));
}

@Test public void doubleMax(){
    double d1 = 0.234d;
    double d2 = 0.000023d;

    assertEquals("max(double) did not return the correct result.", d1, c.max(d1, d2));
    assertEquals("max(double) did not return the correct result.", d2, c.max(-d1, d2));
}

@Test public void floatMax(){
    float f1 = 0.34f;
    float f2 = 0.11f;

    assertEquals("max(float) did not return the correct result.", f1, c.max(f1, f2));
    assertEquals("max(float) did not return the correct result.", f2, c.max(-f1, f2));
}

@Test public void longMin(){
    long l1 = 9999l;
    long l2 = 1111l;

    assertEquals("min(long)␣did␣not␣return␣the␣correct␣result.", l2, c.min(l1, l2));
    assertEquals("min(long)␣did␣not␣return␣the␣correct␣result.", -l1, c.min(-l1, l2));
}

@Test public void intMin(){
    int i1 = 234;
    int i2 = 123;

    assertEquals("min(int)␣did␣not␣return␣the␣correct␣result.", i2, c.min(i1, i2));
    assertEquals("min(int)␣did␣not␣return␣the␣correct␣result.", -i1, c.min(-i1, i2));
}

@Test public void doubleMin(){
    double d1 = 0.234d;
    double d2 = 0.000023d;

    assertEquals("min(double)␣did␣not␣return␣the␣correct␣result.", d2, c.min(d1, d2));
    assertEquals("min(double)␣did␣not␣return␣the␣correct␣result.", -d1, c.min(-d1, d2));
}

@Test public void floatMin(){
    float f1 = 0.34f;
    float f2 = 0.11f;

    assertEquals("min(float)␣did␣not␣return␣the␣correct␣result.", f2, c.min(f1, f2));
    assertEquals("min(float)␣did␣not␣return␣the␣correct␣result.", -f1, c.min(-f1, f2));
}
```