Institut für Informatik
Jun.-Prof. Dr.-Ing. Elke Pulvermüller
B. Sc. Sergey Krutikov
B. Sc. Alexander Brandt

Universität Osnabrück, 08.01.2013
http://www-lehre.inf.uos.de/~sq

Testat bis Freitag, 18.01.2013, 19:00 Uhr

Übungen zu Software-Qualität

Wintersemester 2012/2013

Übungsblatt 11

Aufgabe 11.1 (Mutationen-Test, 26 Punkte)

Machen Sie sich mit der (modifizierten) Methode `pow2()` auf der Rückseite des Aufgabenblattes vertraut. Sie finden die Methode auch in in der JAVA-Klasse unter der folgenden Internet-Adresse:

`http://www-lehre.inf.uos.de/~sq/2012/aufgaben/blatt11/Pow2.java`

a) Geben Sie die unterschiedlichen Mutanten zu dem gegebenen Programm an, für deren Erzeugung die folgende Regel zur Mutationstransformation angewendet werden soll: (6 Punkte)

- Zu allen literalen Zahlenkonstanten wird der Wert 1 addiert.

Dabei ist zu beachten, dass pro Mutant nur eine Modifikation durchgeführt werden soll!

b) Gegeben sind die folgenden Testfallsätze, erzeugt mit jeweils unterschiedlichen Testverfahren:

- Testverfahren A: $x = -2$, $n = 3$
- Testverfahren B: $x = 3$, $n = -4$
- Testverfahren C: $x = -2$, $n = 0$

Begründen Sie, welches der Testverfahren das beste und welches das schlechteste ist. Zeigen Sie hierfür mit dem *starken* Mutationen-Test, welches Testverfahren welche Mutanten erkennt. (9 Punkte)

c) Berechnen Sie für die Testverfahren A, B und C jeweils den Score für den *starken* Mutationen-Test. Was sagen die Werte jeweils aus? (7 Punkte)

d) Geben Sie ein beliebiges Programm und die entsprechende Mutationstransformation an (nicht das Beispiel aus der Vorlesung und/oder der begleitenden Literatur), mit der sich ein *äquivalenter* Mutant erzeugen lässt. Begründen Sie, warum der Mutant äquivalent zum usprünglichen Programm ist. (4 Punkte)

– bitte wenden –

Aufgabe 11.2 (Stilanalyse, 4 Punkte)

Machen Sie sich mit den *Java Code Conventions* von Sun vertraut:

`http://www.oracle.com/technetwork/java/index-135089.html`

Suchen Sie sich zwei beliebige Konventionen heraus (Kapitel 3 bis einschließlich 10) und geben Sie ein kurze Einschätzung an (maximal drei Sätze), warum die jeweilige Konvention existiert, ob und warum sie sinvoll ist und/oder wie alternativ verfahren werden könnte. (4 Punkte)

Aufgabe 11.3 (Slicing, 22 Punkte)

Geben Sie den für das Slicing benötigten Kontrollflussgraphen für die Methode `pow2()` an. (4 Punkte)

a) Führen Sie für die Variablen `i` und `res` jeweils ein (statisches) *Backward-Slicing* durch, ausgehend von der letzten Referenz der jeweiligen Variable. (12 Punkte)

b) Führen Sie für die Variable `i` ein (statisches) *Forward-Slicing* durch, ausgehend von der ersten Definition der Variable. (6 Punkte)

Aufgabe 11.4 (Offener Frageteil, 10 Punkte)

Beantworten Sie Ihrer Tutorin bzw. Ihrem Tutor Fragen zur Veranstaltung „Software-Qualität“.

```
01    /**
02     * Berechnung der n-ten Potenz.
03     *
04     * @param x Basis
05     * @param n Exponent
06     * @return n-te Potenz zur Basis x
07     */
08    public static double pow2(double x, int n) {
09
10        double res;
11        int i;
12
13        i = n;
14        if (n < 0) {
15            i = -n;
16        }
17
18        res = 1.0;
19        while (i > 0) {
20            res = res * x;
21            i = i - 1;
22        }
23
24        if (n < 0) {
25            res = 1 / res;
26        }
27
28        return res;
29    }
```