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

Universität Osnabrück, 13.11.2012
http://www-lehre.inf.uos.de/~sq

Testat bis Freitag, 23.11.2012, 19:00 Uhr

Übungen zu Software-Qualität

Wintersemester 2012/2013

Übungsblatt 5

Aufgabe 5.1 (Testen von Schleifen, 16 Punkte)

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

`http://www-lehre.inf.uos.de/~sq/2012/aufgaben/blatt05/MatrixScalarMultiplication.java`

a) Erzeugen Sie den Kontrollflussgraphen für die Methode `multMatrixScalar()`. (4 Punkte)

b) Geben Sie für die Methode `multMatrixScalar()` die Äquivalenzklassen für den modifizierten *Boundary-Interior*-Test ($k = 2$) an. (10 Punkte)

c) Wieviele Testfälle sind mindestens erforderlich, um die Methode `multMatrixScalar()` entsprechend dem Testkriterium des modifizierten *Boundary Interior Coverage* hinreichend zu testen? (1 Punkt)

d) Wie lautet der allgemeinere Name für das von Ihnen angewendete Testverfahren? (1 Punkt)

Aufgabe 5.2 (Bedingungsüberdeckungstest, 14 Punkte)

Erstellen Sie die Wahrheitstabellen für vollständige und unvollständige Evaluation (4 Punkte) für die folgende Entscheidung:

((A && B) || (C && D))

Finden Sie einen hinreichenden Satz von möglichst sinnvollen Testfällen, wobei Sie die Anzahl der Testfälle gering halten sollen, so dass ...

a) eine einfache Bindungsüberdeckung (engl. *Simple Condition Coverage*) mit vollständiger Evaluation) erreicht wird (begründen Sie Ihre Wahl) (2 Punkte),

b) eine einfache Bindungsüberdeckung (engl. *Simple Condition Coverage*) mit unvollständiger Evaluation erreicht wird (begründen Sie Ihre Wahl) (2 Punkte),

c) eine Bedingungs-/Entscheidungsüberdeckung (engl. *Condition/Decision Coverage*) mit vollständiger Evalution erreicht wird (begründen Sie Ihre Wahl) (2 Punkte),

d) eine Bedingungs-/Entscheidungsüberdeckung (engl. *Condition/Decision Coverage*) mit unvollständiger Evaluation erreicht wird (begründen Sie Ihre Wahl) (2 Punkte).

e) Begründen Sie an Ihrem Beispiel, warum a) das akzeptierte Minimalkriterium der Zweigüberdeckung nicht erreicht. (2 Punkte)

– bitte wenden –

Aufgabe 5.3 (Bedingungsüberdeckungstest, 10 Punkte)

Finden Sie einen hinreichenden Satz von möglichst sinnvollen Testfällen, wobei Sie die Anzahl der Testfälle gering halten sollen, für die Entscheidung

((A && B) || (C && D)),

so dass ...

a) eine minimale Mehrfach-Bedingungsüberdeckung (engl. *Minimal Multiple Condition Coverage*) mit vollständiger Evaluation erreicht wird (begründen Sie Ihre Wahl), (2 Punkte)

b) eine minimale Mehrfach-Bedingungsüberdeckung (engl. *Minimal Multiple Condition Coverage*) mit unvollständiger Evaluation erreicht wird (begründen Sie Ihre Wahl), (2 Punkte)

c) eine modifizierte Bedingungs-/Entscheidungsüberdeckung (engl. *Modified Condition/Decision Coverage*) mit vollständiger Evalution erreicht wird (begründen Sie Ihre Wahl), (2 Punkte)

d) eine modifizierte Bedingungs-/Entscheidungsüberdeckung (engl. *Modified Condition/Decision Coverage*) mit unvollständiger Evaluation erreicht wird (begründen Sie Ihre Wahl), (2 Punkte)

e) eine Mehrfach-Bedingungsüberdeckung (engl. *Multiple Condition Coverage*) mit vollständiger Evalution erreicht wird (begründen Sie Ihre Wahl), (1 Punkt)

f) eine Mehrfach-Bedingungsüberdeckung (engl. *Multiple Condition Coverage*) mit unvollständiger Evaluation erreicht wird (begründen Sie Ihre Wahl). (1 Punkt)

```
01    /**
02     * Method to multiplicate a given input matrix with a given scalar. Returns
03     * the result in a given output matrix. The output matrix has to be of the
04     * same or greater dimensions as/than the input matrix.
05     *
06     * @param in_iScalar scalar to multiplicate the matrix
07     * @param in_aiMatrixA matrix to multiplicate
08     * @param out_aiMatrixB matrix containing the result of the multiplication
09     * @return 'true' if multiplication was successful, 'false' otherwise
10     */
11    public static boolean multMatrixScalar(int in_iScalar, int[][] in_aiMatrixA,
12            int[][] out_aiMatrixB) {
13        boolean bResult = false;
14        // check if the given matrices are not 'null' and the output matrix has
15       // the same or greater dimensions as/than the input matrix
16        if (in_aiMatrixA != null && out_aiMatrixB != null
17                && greaterOrEqualDimensions(in_aiMatrixA, out_aiMatrixB)) {
18            // go through the lines of the input matrix
19            for (int i = 0; i < in_aiMatrixA.length; i++) {
20                // go through the columns of the input matrix and multiplicate
21                for (int j = 0; j < in_aiMatrixA[i].length; j++) {
22                    out_aiMatrixB[i][j] = in_iScalar * in_aiMatrixA[i][j];
23                }
24            }
25            bResult = true;
26        }
27        return bResult;
28    }
```