

Linguaggi di Programmazione per la Sicurezza

Esame del 19 gennaio 2011 - (Prova di laboratorio)

1 Quadrato magico [pt. 2]

Scrivere una classe *QuadratoMagico3* che rappresenti un quadrato 3×3 ; tale classe deve permettere di riempire il quadrato con dei valori e verificare se il quadrato è magico.

All'inizio il quadrato deve essere vuoto (con tutte le celle a 0).

Metodo set La classe deve disporre di un metodo *set(int i, int j, int value)* che setta la cella identificata dagli indici (i, j) a *value*. Il metodo non deve eseguire alcun controllo sul fatto che gli indici siano corretti (compresi tra 0 e 2) e sul fatto che il valore sia corretto (compreso tra 1 e 9).

Metodo isMagic La classe deve disporre di un metodo booleano *isMagic()* che dice se il quadrato è magico. Un quadrato 3×3 è magico se tutte le righe, tutte le colonne e le due diagonali sommano a 15¹.

2 Junit

- In JUnit, per il metodo *set*, scrivere:
 - un caso di test in cui si mostri che la modifica di una cella vuota viene eseguita correttamente; [pt. 1]
 - un caso di test in cui si mostri che la modifica di una cella piena viene eseguita in modo errato. Il caso di test si aspetta che, se si prova a modificare una cella piena, la cella non dovrebbe essere modificata: invece il metodo la modifica. [pt. 1]

¹Un quadrato magico di ordine n contenente tutti gli interi da 1 a n^2 è detto *perfetto* o *normale*. La costante magica di questi quadrati è data dalla formula:

$$M(N) = \frac{1}{n} \sum_{k=1}^{n^2} k = \frac{1}{2} n(n^2 + 1)$$

- scrivere in JUnit un caso di test che evidenzi che è possibile costruire un quadrato che sia magico; nella tabella 1 viene mostrato un quadrato magico. **[pt. 1]**

2	7	6
9	5	1
4	3	8

Table 1: Esempio di quadrato magico 3×3

- scrivere in JUnit un caso di test che evidenzi che è possibile costruire un quadrato che non sia magico. **[pt. 1]**

3 JML

Scrivere in JML le seguenti precondizioni al metodo *set*:

- l'indice i e l'indice j sono indici validi e la cella identificata da (i, j) è vuota; **[pt. 1.5]**
- il valore *value* è un valore valido; **[pt. 0.5]**
- il valore *value* non è già presente nel quadrato. **[pt. 2]**

Totale punti = 2 + 4 + 4 = 10

package esami;

```
public class QuadratoMagico3_20110119 {
    /*@ spec_public @*/ int[][] square;

    public QuadratoMagico3_20110119() {
        square = new int[3][3];
    }

    //Scrivere in JML le seguenti precondizioni al metodo set

    //l'indice i e l'indice j sono indici validi e la cella identificata da (i, j) e' vuota
    //@ requires (h >= 0 && h <= 2) && (k >= 0 && k <= 2) && square[h][k] == 0;

    //il valore value e' un valore valido
    //@ requires value >= 0 && value <= 9;

    //il valore value non e' gia' presente nel quadrato
    //@ requires !(exists int i, j; i >= 0 && i <= 2 && j >= 0 && j <= 2; square[i][j] == value);
    //oppure
    //@ requires (forall int i, j; i >= 0 && i <= 2 && j >= 0 && j <= 2; square[i][j] != value);
    public void set(int h, int k, int value) {
        square[h][k] = value;
    }

    public boolean isMagic() {
        int sum;
        for(int i = 0; i < 3; i++) {
            sum = 0;
            for(int j = 0; j < 3; j++) {
                sum = sum + square[i][j];
            }
            if(sum != 15) {
                return false;
            }
        }
        for(int j = 0; j < 3; j++) {
            sum = 0;
            for(int i = 0; i < 3; i++) {
                sum = sum + square[i][j];
            }
            if(sum != 15) {
                return false;
            }
        }
        if(square[0][0] + square[1][1] + square[2][2] != 15) {
            return false;
        }
        if(square[0][2] + square[1][1] + square[2][0] != 15) {
            return false;
        }
        return true;
    }

    public static void main(String[] args) {
        QuadratoMagico3_20110119 q = new QuadratoMagico3_20110119();
        q.set(0, 0, 1);
    }
}
```

```
q.set(0, 1, 2);  
//q.set(0, 1, 3);//viola preconditione  
//q.set(3, 1, 9);//viola preconditione  
//q.set(2, 2, 10);//viola preconditione  
}  
}
```

```

package esami;

import static org.junit.Assert.*;

import org.junit.Test;

public class Test_QuadratoMagico3_20110119 {

    //caso di test in cui si mostra che la modifica di una cella vuota viene
    //eseguita correttamente
    @Test
    public void testSet1() {
        QuadratoMagico3_20110119 q = new QuadratoMagico3_20110119();
        q.set(0, 0, 1);
        assertEquals(1, q.square[0][0]);
    }

    //caso di test in cui si mostra che la modifica di una cella piena viene
    //eseguita in modo errato. Il caso di test si aspetta che, se si prova a modificare
    //una cella piena, la cella non dovrebbe essere modificata: invece il metodo
    //la modifica.
    @Test
    public void testSet2() {
        QuadratoMagico3_20110119 q = new QuadratoMagico3_20110119();
        q.set(0, 0, 1);
        q.set(0, 0, 2);
        assertEquals(1, q.square[0][0]);
    }

    //caso di test che evidenzia che e' possibile costruire un quadrato
    //che sia magico
    @Test
    public void testIsMagic1() {
        QuadratoMagico3_20110119 q = new QuadratoMagico3_20110119();
        q.set(0, 0, 2);
        q.set(0, 1, 7);
        q.set(0, 2, 6);
        q.set(1, 0, 9);
        q.set(1, 1, 5);
        q.set(1, 2, 1);
        q.set(2, 0, 4);
        q.set(2, 1, 3);
        q.set(2, 2, 8);
        assertTrue(q.isMagic());
    }

    //caso di test che evidenzia che e' possibile costruire un quadrato
    //che non sia magico
    @Test
    public void testIsMagic2() {
        QuadratoMagico3_20110119 q = new QuadratoMagico3_20110119();
        int value = 1;
        for(int i = 0; i < 3; i++) {
            for(int j = 0; j < 3; j++) {
                q.set(i, j, value);
                value++;
            }
        }
    }
}

```

```
    }  
    assertFalse(q.isMagic());  
  }  
}
```