**СОФТВЕРСКИ КВАЛИТЕТ И ТЕСТИРАЊЕ**

**ДОМАШНА ЗАДАЧА 3**

Фисник Лимани, 151027

**Дадена ни е функцијата:**

public Object findFirstLargerElementThan(List list, Object el)

{

// Effects: If list or element is null throw NullPointerException

// else if there is larger element than el in the list return the first larger

// element

// else if there is no larger element than el return null

}

**INPUT DOMAIN MODELING**

1. **INTERFACE-BASED APPROACH**

Карактеристики:

1. состојбата на list? null, empty, 1 or more elements
2. el е null? True, False
3. **FUNCTIONALITY-BASED APPROACH**

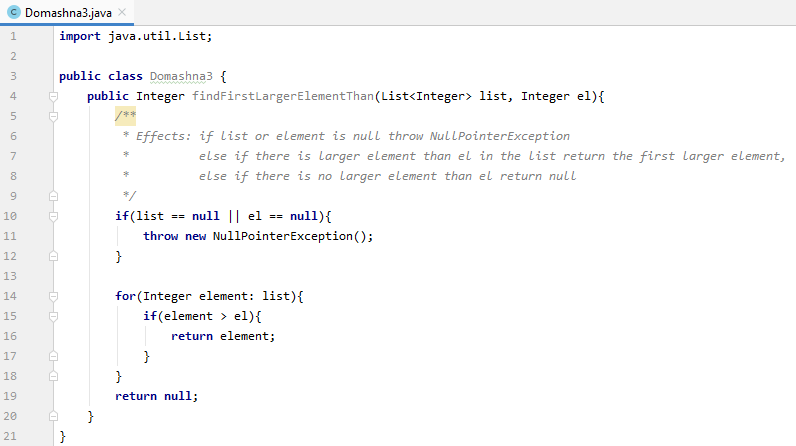
Карактеристики:

1. Дали има елемент поголем од el во листата? True, False
2. Позицијата на првиот елемент поголем од el во листата?

* Елементот е на првата позиција
* Елементот е на последната позиција
* Елементот е на некоја друга позиција, освен првата и последната
* Таа позиција не постои

1. Партиционирањето на влезните параметри го задоволуваат својството на дисјунктност бидејќи нема ниеден случај каде е можно една вредност да припаѓа на два блокови под истата карактеристика.
2. Партиционирањето на влезните параметри го задоволуваат својството на комплетност бидејќи нема случај кога не е опфатена некоја состојба под која може да се најде една карактеристика.
3. **Base Choice Coverage (BCC)**
4. Interface-Based пристапот
   * Base choices (црвени):
     + state of list? null, empty, **1 or more elements**
     + el is null? True, **False**
   * Тестови:
     + **1 or more elements, False**
     + **1 or more elements,** True
     + Null**, False**
     + Empty**, False**
   * Бројот на тестови (според формулата):
     + 1
     + + (БројНаБлоковиЗаКарактеристика1 – 1)
     + + (БројНаБлоковиЗаКарактеристика1 – 1)
     + = 1 + (3 – 1) + (2 – 1) = 1 + 2 + 1 = 4 тестови
5. Functional-Based пристапот
   * Base choices (црвени):
     + C1: Дали има елемент поголем од el во листата?
       - **B1: True**
       - B2: False
     + C2: Позицијата на првиот елемент поголем од el во листата?
       - B1: Елементот е на првата позиција
       - B2: Елементот е на последната позиција
       - **B3:** Елементот е на некоја друга позиција, освен првата и последната
       - B4: Таа позиција не постои
   * Tests:
     + C1.B1, C2.B3
     + C1.B1, C2.B1
     + C1.B1, C2.B2
     + ~~C1.B1, C2.B4~~
       - INFEASIBLE – не е можно да постои таков елемент (од C1.B1) а да не му го знаеме позицијата/индексот (од C2.B4)
     + ~~C1.B2, C2.B3~~
       - INFEASIBLE – не е можно да не постои таков елемент (од C1.B2), а позицијата на првиот елемент поголем од el да му е некоја позиција од 2 до n – 2 (каде n е бројот на елементите во листата)
   * Бројот на тестови:
     + Според формулата:
       - 1 + (2-1) + (4-1) = 1 + 1 + 3 = 5
     + 2 од тестовите се INFEASIBLE
       - Значи, 5 – 2 = 3 тестови
6. **Junit тестови**

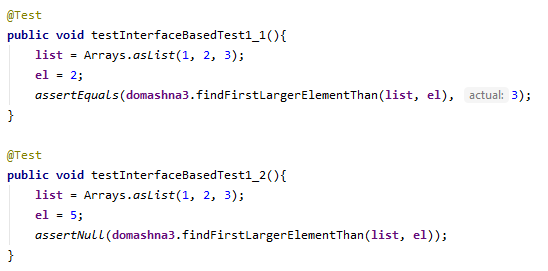
Прво ќе го имплементираме функцијата, и таа имплементација ќе може да се види во следната слика:

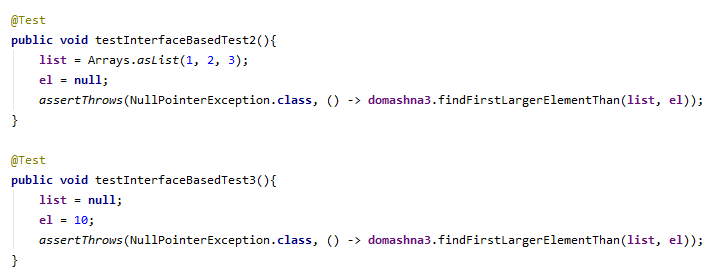


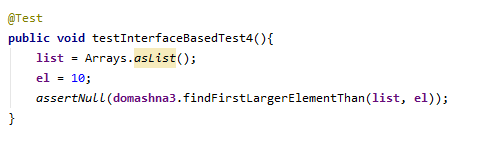
**Конкретни тестови:**

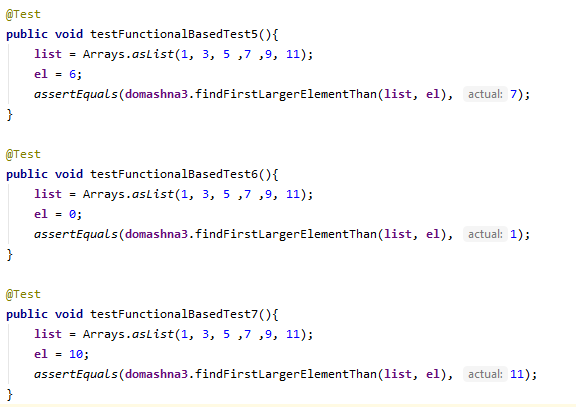
|  |  |  |
| --- | --- | --- |
| Тестови | list | el |
| Тест1 (Interface-Based) | [1,2,3] | 2 |
|  | [1,2,3] | 5 |
| Тест2 (Interface-Based) | [1,2,3] | null |
| Тест3 (Interface-Based) | null | 10 |
| Тест4 (Interface-Based) | [] | 10 |
|  |  |  |
| Тест5 (Functional-Based) | [1,3,5,7,9,11] | 6 |
| Тест6 (Functional-Based | [1,3,5,7,9,11] | 0 |
| Тест7 (Functional-Based | [1,3,5,7,9,11] | 10 |

**Кодови:**









Резултатите од тестирањата:

