

Софтверски квалитет и тестирање

Домашна задача 1 и 2

Ема Спирова

1. Креирање тестови за дадената функција

Функција palindrome:

```
public class Palindrom {

    public Boolean palindrome (String word){
        for(int i = 0; i < word.length(); i++) {

if(word.toLowerCase().charAt(i) != word.toLowerCase().charAt(word.length()-1 -i)){

                return false;
            }
        }
        return true;
    }

}
```

За функцијата palindrome ги напишав следните тестови:

```
import static org.junit.jupiter.api.Assertions.*;

class Testovi {

    @Test
    public void test() {
        Palindrom palin = new Palindrom();
        boolean output = palin.palindrome("racecar");
        assertEquals(true, output);
    }

    @Test
    public void test2() {
        Palindrom palin = new Palindrom();
        boolean output = palin.palindrome("test");
        assertEquals(false, output);
    }

    @Test
    public void test3() {
        Palindrom palin = new Palindrom();
        boolean output = palin.palindrome("testc@se");
    }

}
```

```

        assertEquals(false, output);
    }

    @Test
    public void test4() {
        Palindrom palin = new Palindrom();
        boolean output = palin.palindrome("151");
        assertEquals(true, output);
    }

}

```

2. Одговорете дали со креираните тестови се откриваат можните недостатоци и испади на напишаната функција.

- Не се откриваат целосно можните недостатоци и испади на функцијата `palindrome`. Во барањето со точка 3 се додадени тестови ако: стрингот е составен од една буква, две букви (палиндром и непалиндром), исто така за да не се прави разлика помеѓу голема и мала буква, ако е сепак палиндром зборот, е додаден уште еден тест.

3. Поправете ги тестовите за да ги откриете недостатоците и испадите на напишаната функција.

```

import static org.junit.jupiter.api.Assertions.*;

class Testovi {

    @Test
    public void test5() {
        Palindrom palin = new Palindrom();
        boolean output = palin.palindrome("eyE");
        assertEquals(true, output);
    }

    @Test
    public void shouldRecognizeOneCharacterPalindrome() {
        Palindrom palin = new Palindrom();
        boolean output = palin.palindrome("a");
        assertEquals(true, output);
    }

    @Test
    public void shouldRecognizeTwoCharacterPalindrome() {
        Palindrom palin = new Palindrom();
        boolean output = palin.palindrome("bb");
        assertEquals(true, output);
    }

}

```

```

@Test
public void shouldNotRecognizeTwoCharacterPalindrome() {
    Palindrom palin = new Palindrom();
    boolean output = palin.palindrome("ab");
    assertEquals(false, output);
}

}

```

4. Параметризирани тестови.

```

import static org.junit.jupiter.api.Assertions.*;
import static org.junit.Assert.assertEquals;
import java.util.Arrays;
import java.util.Collection;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.params.ParameterizedTest;
import org.junit.jupiter.params.provider.MethodSource;

class PalindromTest {

    private Palindrom palindromeChecker;

    @BeforeEach
    public void setup(){
        palindromeChecker = new Palindrom();
    }

    public static Collection<Object[]> palindromi() {
        return Arrays.asList(new Object[][] {
            { "racecar", true },
            { "Cdkjd", false },
            { "ANA", true },
            { "2#test", false },
            { "level", true },
            { "Madam", true }
        });
    }

    @ParameterizedTest
    @MethodSource("palindromi")

    public void testPrimeNumberChecker (String wordInput, boolean expectedResult) {
        System.out.println("Parameterized Word is : "+ wordInput);
        assertEquals(expectedResult, palindromeChecker.palindrome(wordInput));
    }

}

```

5. Креирање тестови за исклучоци.

- Исклучок ако се работи за празен String
- Исклучок за null String

Дополнување во класата Palindrom:

```
public class Palindrom {

    public Boolean palindrome (String word) {
        for(int i = 0; i < word.length(); i++) {
            if(word.toLowerCase().charAt(i) != word.toLowerCase().charAt(word.length()-1 -i)){
                return false;
            }
        }

        return true;
    }

    //isklucok za prazen String
    public void empty(String word) throws Exception{
        if(word == "") {
            throw new Exception("Empty String");
        }
    }

    //isklucok za null String
    public void nullString(String word) {
        if(word == null) {
            throw new NullPointerException();
        }
    }
}
```

Креирање на тестови за исклучоци:

```
class PalindromTestTest {
```

```
@Test
```

```
public void test() {  
    Palindrom prazenString = new Palindrom();  
    assertThrows(Exception.class, () ->prazenString.empty(""));  
}
```

```
@Test
```

```
public void testZaNull() {  
    Palindrom nullStr = new Palindrom();  
    assertThrows(NullPointerException.class, () -> nullStr.nullString(null));  
    }  
}
```