Disciplina: **Teste**

Antonio Carlos Ferreira de Almeida - RA:2438194

Exercício 2

Aberto: segunda, 23 jan 2023, 08:00
Vencimento: domingo, 5 fev 2023, 23:59

1. Verifique a especificação do programa *Identifier:*

   O programa deve determinar se um *Identifier* é válido ou não.
   Um identificador válido deve começar com uma letra e conter apenas letras e/ou dígitos. Além disso, deve ter no mínimo um caractere e no máximo seis caracteres de comprimento.

   Exemplo:
   abc12 (válido);
   cont*1 (inválido);
   1soma (inválido);
   a123456 (inválido).

   Verifique o particionamento em classes de equivalência e um exemplo de conjunto de casos de teste:

Condições de entrada	Classes válidas	Classes inválidas
Tamanho '*t*' do identificador	1 <=*t*<=6 (1) Sim	t > 6 (2) Não
Primeiro caracter '*c*' é uma letra	(3) Sim	(4) Não
Só contém caracteres válidos	(5) Sim	(6) Não

Exemplo de conjunto de casos de teste

t_0

(a1,Válido)	(2B3, Inválido)	(Z-12, Inválido)	(A1b2C3d, Inválido)
(1,3,5)	(4)	(6)	(2)

1. Implemente em Java o método para o programa *Identifier*.
   - O método deverá receber o identificador por passagem de valor.
   - Fazer o uso exceções para o tratamento das classes inválidas.
   - Não é necessário fazer a *View* para a entrada dos dados.
2. Implemente no JUnit os casos de teste conforme o particionamento em classes de equivalência.

Entregue os itens 1 e 2 em um arquivo único .pdf
(coloque o código necessário para responder os itens usando seu editor de texto favorito e salve em um arquivo .pdf para entrega).

Respostas:

01.

App.java | IdentifierSolucion.java | AppTest.java

```java
package br.identifierSolucion;

public class IdentifierSolucion {

    public String validateIdentifier(String varIdentifier) {

        String response = "Ok";
        char cChar;
        int iInt, nInt;

        try {
            nInt = varIdentifier.length();
            for (iInt = 0; iInt < nInt; iInt++) {
                cChar = varIdentifier.charAt(iInt);
                if (nInt > 6) {
                    response = "noOk";
                    throw new Exception("...Identifier's > 6 ");
                }
                if (!Character.isLetter(varIdentifier.charAt(0))) {
                    response = "noOk";
                    throw new Exception("...First caracter isn't letter ");
                }
                if (!(Character.isLetter(cChar) || Character.isDigit(cChar))) {
                    response = "noOk";
                    throw new Exception("...There is invalid character ");
                }
            }
            System.out.println("Perfect...");
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
        return response;
    }
}
```

2.

App.java | IdentifierSolucion.java | *AppTest.java

```java
package br.identifierTest;

import static org.junit.jupiter.api.Assertions.assertEquals;
import org.junit.jupiter.api.Test;
import br.identifierSolucion.IdentifierSolucion;

public class AppTest {

    String test = "";
    String tamanhoDoIdentificadorOk = "a1de5a";
    String tamanhoDoIdentificadorNoOk = "a1deqaa";
    String primeiroCaracterLetraOk = "a1deq";
    String primeiroCaracterLetraNoOk = "1a1deq";
    String soContemCaracteresValidosOk = "a1de5q";
    String soContemCaracteresValidosNoOk = "a1^*@q";

    @Test
    void sizeIdentifierOk() {
        test = new IdentifierSolucion().validateIdentifier(tamanhoDoIdentificadorOk);
        assertEquals("Ok", test);
    }

```

```
22
23      @Test
24      void sizeIdentifierInvalidate() {
25          test = new IdentifierSolucion().validateIdentifier(tamanhoDoIdentificadorNoOk);
26          assertEquals("noOk", test);
27      }
28
29      @Test
30      void identifierFirstCharacterOk() {
31          test = new IdentifierSolucion().validateIdentifier(primeiroCaracterLetraOk);
32          assertEquals("Ok", test);
33      }
34
35      @Test
36      void identifierFirstCharacterInvalidate() {
37          test = new IdentifierSolucion().validateIdentifier(primeiroCaracterLetraNoOk);
38          assertEquals("noOk", test);
39      }
40
41      @Test
42      void identifierOwnCharacterOk() {
43          test = new IdentifierSolucion().validateIdentifier(soContemCaracteresValidosOk);
44          assertEquals("Ok", test);
45      }
46
47      @Test
48      void identifierNoContainCharacterValid() {
49          test = new IdentifierSolucion().validateIdentifier(soContemCaracteresValidosNoOk);
50          assertEquals("noOk", test);
51      }
52  }
```