```
public void testCalculaSalari() {
 float resultatReal;
 try {
  resultatReal = Empleat.calculaSalari(2000.0, 8.0);
  double resultatEsperat = 1360.0;
  assertEquals(resultatEsperat, resultatReal, 0.01);
 } catch (MIException e) {
  fail("Llançada excepció no esperada MIException");
 }
}
```

Si el mètode provat llança una excepció no marcada (unchecked) que no hem controlat, llavors ho indica com un error no com una fallada de la prova.

Exemple

La classe Matricad **conté uns mètodes que treballen sobre una llista de cadenes.**

```
public class Matricad {
   private java.util.ArrayList<String> cadenes;                // referència a la llista de cadenes, un camp
   /**
    * Constructor de Matricad.
    * @param dada matriu amb les cadenes per a la llista
    */
   public Matricad(String[] dada) {
      if ((dada == null) || (dada.length == 0)) {.             // Verifiquem que la llista tinga valors
         throw new IllegalArgumentException();
      }
      this.cadenes = new java.util.ArrayList<>();
      for (String element : dada) {
         cadenes.add(element);
      }
   }
   /**
    * @return la cadena que té més caràcters. La primera si hi ha diverses amb la mateixa longitud
    */
   public String getMaxCad() {
      String max = "";
      for (String element : cadenes) {
         if (element.length() > max.length()) {
            max = element;
         }
      }
      return max;
   }
   /**
    * @return la suma total de caràcters de totes les cadenes.
    */
   public int getSumaCaracters() {
      int total = 0;
      for (String d : cadenes) {
         total += d.length();
      }
      return total;
```

```
  }
  /**
   * Retorna l'índex de la cadena buscada.
   *
   * @param unaCadena Cadena buscada
   * @return Retorna la posició d'un element dins de l'array
   * @throws java.util.NoSuchElementException Si l'element no existeix en la llista
   */
  public int getIndexDe(String unaCadena)
          throws java.util.NoSuchElementException {
    if (unaCadena == null) {                          // Comprovem que l'argument siga vàlid
      throw new IllegalArgumentException();
    }
    int pos = 0;
    for (String d : cadenes) {                        // Recorrem la informació fins a trobar l'element
      if (d.equals(unaCadena)) {
        return pos;
      }
      pos++;
    }
    throw new java.util.NoSuchElementException(unaCadena);      // L'element no existeix, llancem l'excepció
  }
}
```

Quan es crea la unitat de proves, es marquen totes les opcions. Per a TestNG es genera la classe de proves següent

```
public class MatriCadNGTest {
  public MatriCadNGTest() {
  }
  @BeforeClass
  public static void setUpClass() throws Exception {        // s'executa una única vegada a l'inici de la prova
  }
  @AfterClass
  public static void tearDownClass() throws Exception {    // s'executa una única vegada al final de la prova
  }
  @BeforeMethod
  public void setUpMethod() throws Exception {             // s'executa cada vegada a l'inici del mètode
  }
  @AfterMethod
  public void tearDownMethod() throws Exception {          // s'executa cada vegada al final del mètode
  }
  /**
   * Test of getMaxCad method, of class MatriCad.
   */
  @Test
  public void testGetMaxCad() {
    System.out.println("getMaxCad");
    MatriCad instance = null;
    String expResult = "";
    String result = instance.getMaxCad();
    assertEquals(result, expResult);
    // TODO review the generated test code and remove the default call to fail.
    fail("The test case is a prototype.");
```

```
    }
    /**
     * Test of getSumaCaracters method, of class MatriCad.
     */
    @Test
    public void testGetSumaCaracters() {
        System.out.println("getSumaCaracters");
        MatriCad instance = null;
        int expResult = 0;
        int result = instance.getSumaCaracters();
        assertEquals(result, expResult);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }
    /**
     * Test of getIndexDe method, of class MatriCad.
     */
    @Test
    public void testGetIndexDe() {
        System.out.println("getIndexDe");
        String unaCadena = "";
        MatriCad instance = null;
        int expResult = 0;
        int result = instance.getIndexDe(unaCadena);
        assertEquals(result, expResult);
        // TODO review the generated test code and remove the default call to fail.
        fail("The test case is a prototype.");
    }
}
```

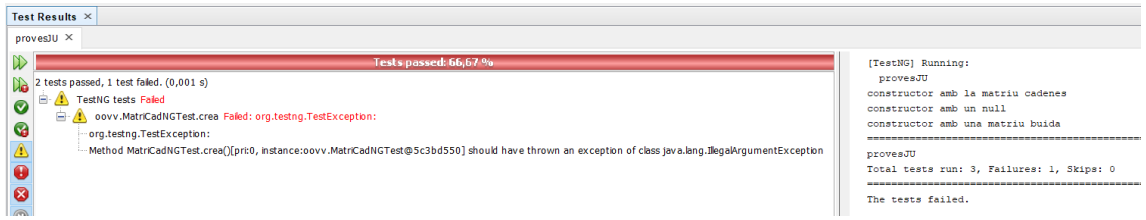
Modifiquem la classe de proves, afegim el camp estàtic cadenes **que contindrà una matriu de cadenes. El camp és estàtic perquè es va a usar en el mètode** setUpClass **que és estàtic.**

S'eliminen els mètodes tearDownClass**,** setUpMethod **i** tearDownMethod**, ja que no anem a usar-los.**

Anem a fer tres proves al constructor, amb la matriu cadenes**, un** null **i una matriu buida, s'ha de llançar l'excepció. La primera prova fallarà.**

```
public class MatriCadNGTest {
    public MatriCadNGTest() {
    }
    static String[] cadenes;      //matriu de cadenes
    @BeforeClass
    public static void setUpClass() throws Exception {
        cadenes = new String[]{"hui", "és", "dilluns", "i", "no", "m'agrada", "gens"};      // carrega la matriu de cadenes
    }
    /**
     * Test del constructor de la classe MatriCad. comprova que es llança
     * IllegalArgumentException si es crea amb una matriu de cadenes
     */
    @Test(expectedExceptions = {java.lang.IllegalArgumentException.class})
    public void crea() {
```

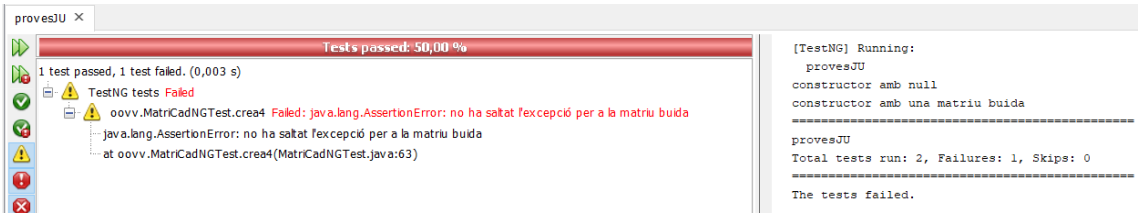
```
    System.out.println("constructor amb la matriu cadenes");
    MatriCad instance = new MatriCad(cadenes);            // falla, ja que no llança l'excepció
  }
  /**
   * Test del constructor de la classe MatriCad. comprova que es llança
   * IllegalArgumentException si es crea amb un null
   */
  @Test(expectedExceptions = {java.lang.IllegalArgumentException.class})
  public void crea1() {
    System.out.println("constructor amb un null");
    MatriCad instance = new MatriCad(null);
  }
  /**
   * Test del constructor de la classe MatriCad. comprova que es llança
   * IllegalArgumentException si es crea amb una matriu buida
   */
  @Test(expectedExceptions = {java.lang.IllegalArgumentException.class})
  public void crea2() {
    System.out.println("constructor amb una matriu buida");
    String[] cads = {};
    MatriCad instance = new MatriCad(cads);
  }
}
```



La comprovació de la creació de Matricad **amb un** null **o amb una matriu buida usant el** fail **és la següent**

```
@Test
public void crea3() {
  try {
    System.out.println("constructor amb null");
    MatriCad instance = new MatriCad (null);     // llança l'excepció i salta al catch i no executa el fail
    fail("no ha saltat l'excepció per a null");
  } catch (IllegalArgumentException e) {
  }
}
@Test
public void crea4() {
  try {
    System.out.println("constructor amb una matriu buida");
    String[] mat = {""};        // la matriu conté una cadena buida, la matriu no està buida
    MatriCadinstance = new MatriCad (mat);                // no llança l'excepció
    fail("no ha saltat l'excepció per a la matriu buida");     // executa el fail
  } catch (IllegalArgumentException e) {
  }
}
```

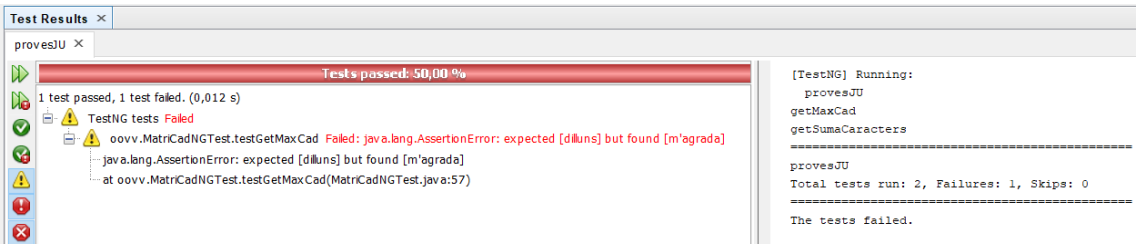
La segona prova falla.



Provem els mètodes getMaxCad **i** getSumaCaracters**, la primera prova falla ja que "dilluns" no és la paraula més llarga.**

```
public class MatriCadNGTest {
    public MatriCadNGTest() {
    }
    static String[] cadenes;      //matriu de cadenes
    @BeforeClass
    public static void setUpClass() throws Exception {
        cadenes = new String[]{"hui", "és", "dilluns", "i", "no", "m'agrada", "gens"};      // carrega la matriu de cadenes
    }
    /**
     * Test of getMaxCad method, of class MatriCad. comprova que "dilluns" és la paraula més llarga
     */
    @Test
    public void testGetMaxCad() {
        System.out.println("getMaxCad");
        MatriCad instance = new MatriCad(cadenes);
        String expResult = "dilluns";
        String result = instance.getMaxCad();
        assertEquals(result, expResult);
    }
    /**
     * Test of getSumaCaracters method, of class Matricad. comprova que les cadenes de la matriu
     * sumen 27 caràcters, es fixa un temps límit de 25 mil·lisegons per a l'execució del test
     */
    @Test(timeOut = 25)
    public void testGetSumaCaracters() {
        System.out.println("getSumaCaracters");
        MatriCad instance = new MatriCad(cadenes);
        int expResult = 27;
        int result = instance.getSumaCaracters();
        assertEquals(expResult, result);
    }
}
```

En la prova de getSumaCaracters**, si el temps d'execució excedeix de 25 mil·lisegons, llavors la prova falla.**



Anem a fer tres proves sobre el mètode getIndexDe**, en la primera es comprova que la posició és la correcta, en les dos següents s'espera una excepció.**

```
public class MatriCadNGTest {
    public MatriCadNGTest() {
    }
    static String[] cadenes;      //matriu de cadenes
    @BeforeClass
    public static void setUpClass() throws Exception {
        cadenes = new String[]{"hui", "és", "dilluns", "i", "no", "m'agrada", "gens"};      // carrega la matriu de cadenes
    }
    /**
     * Test of getIndexDe method, of class MatriCad. comprova que la posició de "dilluns" és la 2
     */
    @Test
    public void testGetIndexDe() {
        System.out.println("GetIndexOf comprova la posició d'una cadena");
        MatriCad instance = new MatriCad(cadenes);
        int expResult = 2;
        int result = instance.getIndexDe("dilluns");
        assertEquals(expResult, result);
    }
    /**
     * Test of getIndexDe method, of class MatriCad. comprova que es llança
     * IllegalArgumentException si es cerca l'índex de null
     */
    @Test(expectedExceptions = {java.lang.IllegalArgumentException.class})
    public void testGetIndexDe1() {
        System.out.println("GetIndexOf cerca d'un valor nul");
        MatriCad instance = new MatriCad(cadenes);
        instance.getIndexDe(null);
    }
    /**
     * Test of getIndexDe method, of class MatriCad. comprova que es llança
     * NoSuchElementException si es cerca un valor que no està en la matriu
     */
    @Test(expectedExceptions = {java.util.NoSuchElementException.class})
    public void testGetIndexDe2() {
        System.out.println("GetIndexOf cerca d'un valor inexistent");
        MatriCad instance = new MatriCad(cadenes);
        instance.getIndexDe("lunes");
    }
}
```

Els tres funcionen correctament