Corrigé TP exo1 :

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

import org.junit.Test;

import main.Tableau;

public class TableauTest {

	@Test
	public void testMoyennValid() {
		int [] contenu= {1,2,4};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		assertEquals((7/3), t.moyenne(), 0.0001 );
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
	@Test
	public void testMoyennValid2() {
		int [] contenu= {-1,-2,-3,-4,-5};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		assertEquals(-3.0, t.moyenne(), 0.0001);
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
	@Test
	public void testMoyennValid3() {
		int [] contenu= {0,0,0,0,0};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		assertEquals(0, t.moyenne(), 0.0001);
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
	@Test (expected = java.lang.ArithmeticException.class)
	public void testMoyennInvalid() {
		int [] contenu= {};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		t.moyenne();

	}
```

```
	@Test
	public void testMoyennValid4() {
		int [] contenu=
{Integer.MAX_VALUE,Integer.MAX_VALUE};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
	  //  erreur ici parceque normalement on peut pas
faire l'addition de deux MaxValue
		//demander aux étudiants d'améliorer le code
pour eviter cette erreur
		assertEquals(Integer.MAX_VALUE ,
t.moyenne(),0.0001);
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
	@Test
	public void testMoyennValid5() {
		int [] contenu=
{Integer.MIN_VALUE,Integer.MIN_VALUE};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		  //  erreur ici parceque normalement on peut
pas faire l'addition de deux MaxValue
			//demander aux étudiants d'améliorer le
code pour eviter cette erreur

		assertEquals(Integer.MIN_VALUE ,
t.moyenne(),0.0001);
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
```

solution exo2

```
;

public class TableauTest {

	@Test
	public void testValid1() {
		int [] contenu= {5,2,4,1,3};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		assertEquals(2, t.getNValue (2));
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
	@Test
	public void testValid2() {
		int [] contenu= {5,5,5,5,5};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		assertEquals(-1, t.getNValue (2));
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}

----------------------------------------
	@Test (expected = java.lang.ArithmeticException.class)
	public void testInvalid1() {
		int [] contenu= {};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		t.getNValue (2);

	}

	@Test (expected = java.lang.Exception.class)
	public void testInvalid1() {
		int [] contenu= {5,2,4,1,3};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		t.getNValue (0);

	}
@Test (expected = java.lang.Exception.class)
	public void testInvalid1() {
		int [] contenu= {5,2,4,1,3};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		t.getNValue (-1);

	}



@Test (expected = java.lang.Exception.class)
	public void testInvalid1() {
```

```
		int [] contenu= {5,2,4,1,3};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		t.getNValue (6);

	}
```

solution exo3

```
;

public class TableauTest {

	@Test
	public void testValid1() {
		int [] contenu= {2,4,6};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		assertEquals(3, t.occurence());
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
	@Test
	public void testValid1() {
		int [] contenu= {-2,-4,-6};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		assertEquals(3, t.occurence());
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}

	@Test
	public void testValid2() {
		int [] contenu= {1,3,5};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		assertEquals(-1, t.occurence());
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}

----------------------------------------
	@Test (expected = java.lang.ArithmeticException.class)
	public void testInvalid1() {
		int [] contenu= {};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		t.occurence();
	}

	@Test (expected = java.lang.Exception.class)
	public void testInvalid2() {
		int [] contenu= {"1","2"};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
```

```
            t.occurence();
        }
    }
@Test (expected = java.lang.Exception.class)
public void testInvalid2() {
    int [] contenu= {Integer.MAX_VALUE+1, Integer.MAX_VALUE+1};
    int taille = contenu.length;
    Tableau t = new Tableau(taille,contenu);
    t.occurence();
}

@Test (expected = java.lang.Exception.class)
public void testInvalid2() {
    int [] contenu= {Integer.MIN_VALUE-1, Integer.MIN_VALUE-1};
    int taille = contenu.length;
    Tableau t = new Tableau(taille,contenu);
    t.occurence();
}
```

solution exo4

```
;

public class TableauTest {

    @Test
    public void testValid1() {
        int [] contenu= {1,3,5,6,2,4};
        int taille = contenu.length;
        Tableau t = new Tableau(taille,contenu);
        assertEquals(true, t.permutation());
        assertEquals(taille, t.getTaille());
        assertArrayEquals(contenu, t.getContenu());
    }
    @Test
    public void testValid2() {
        int [] contenu= {1,1,5,6,2,4};
        int taille = contenu.length;
        Tableau t = new Tableau(taille,contenu);
        assertEquals(false, t.permutation());
        assertEquals(taille, t.getTaille());
        assertArrayEquals(contenu, t.getContenu());
    }
    @Test
    public void testValid3() {
        int [] contenu= {10,2,1,5};
        int taille = contenu.length;
        Tableau t = new Tableau(taille,contenu);
        assertEquals(false, t.permutation());
        assertEquals(taille, t.getTaille());
        assertArrayEquals(contenu, t.getContenu());
    }
    @Test
    public void testValid4() {
        int [] contenu= {10,10,5};
```

```
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		assertEquals(false, t.permutation());
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}

	@Test
	public void testValid5() {
		int [] contenu= {1};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		assertEquals(true, t.permutation());
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
	@Test
	public void testValid6() {
		int [] contenu= {5};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		assertEquals(false, t.permutation());
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
```

```
	@Test (expected = java.lang.ArithmeticException.class)
	public void testInvalid1() {
		int [] contenu= {};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		t.permutation()
	}

	@Test (expected = java.lang.Exception.class)
	public void testInvalid2() {
		int [] contenu= {"1","2"};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		t.permutation();
	    }
	}
```

solution exo5

```
;

public class TableauTest {

	@Test
	public void testValid1() {
		int [] contenu= {3,2,1,5,4};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		assertEquals(3, t.descente());
```

```
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
	@Test
	public void testValid2() {
		int [] contenu= {10,9,8,7,6};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		assertEquals(4, t.descente());
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
	@Test
	public void testValid3() {
		int [] contenu= {1,2,3,4,5};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		assertEquals(0, t.descente());
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
	@Test
	public void testValid4() {
		int [] contenu= {5};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		assertEquals(-1, t.descente());
		assertEquals(taille, t.getTaille());
		assertArrayEquals(contenu, t.getContenu());
	}
```

```
	@Test (expected = java.lang.ArithmeticException.class)
	public void testInvalid1() {
		int [] contenu= {};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu) ;
		t.descente();
	}

	@Test (expected = java.lang.Exception.class)
	public void testInvalid2() {
		int [] contenu= {"1","2"};
		int taille = contenu.length;
		Tableau t = new Tableau(taille,contenu);
		t.descente();
	    }
	}
```