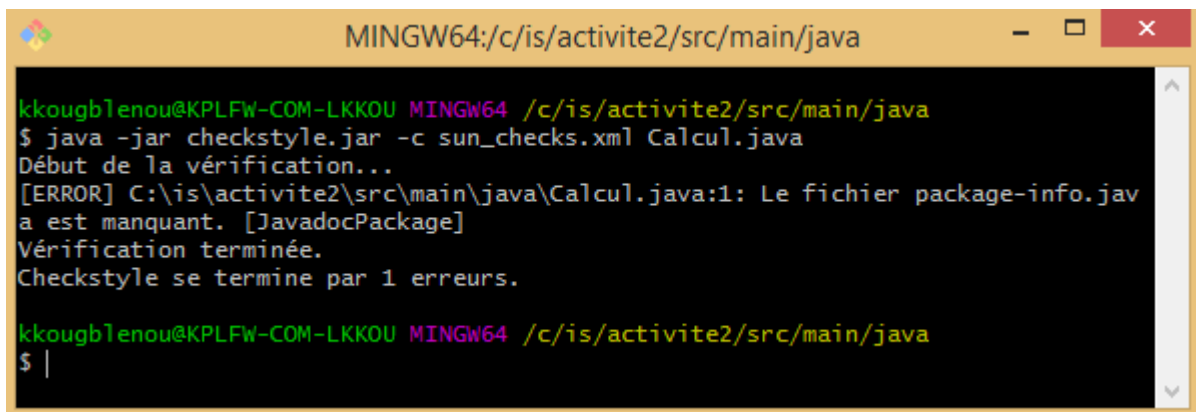


EXERCICE 1 :

1- Veuillez trouver ci-dessous le code réécrit :

```
/** Une classe.
 *
 * @author koffi kougblenou
 */
public final class Calcul {
    /** k est une constant entier. */
    static final int K = 10;
    /** Constructeur de la classe. */
    private void calcul() { }
    /** Calcul la somme de deux nombres.
     * @param a est un entier
     * @param b est un entier
     * @return a + b
     */
    public static int somme(final int a, final int b) {
        return a + b;
    }
    /**
     * @param a est un entier.
     * @param b est un entier
     * @return a / b si b >= 0 et b sinon
     */
    public static int maFonction(final int a, final int b) {
        if (b >= K) {
            return a / b;
        } else {
            return b;
        }
    }
    /**
     * @param a est un entier.
     * @param b est un entier
     * @return a / b si b != 0
     * @throws IllegalArgumentException si b == 0
     */
    public static int division(final int a, final int b) {
        if (b == 0) {
            throw new IllegalArgumentException("b ne doit pas etre 0");
        } else {
            return a / b;
        }
    }
}
```



```
MINGW64:/c:/is/activite2/src/main/java
kkougblenou@KPLFW-COM-LKKOU MINGW64 /c:/is/activite2/src/main/java
$ java -jar checkstyle.jar -c sun_checks.xml Calcul.java
Début de la vérification...
[ERROR] C:\is\activite2\src\main\java\Calcul.java:1: Le fichier package-info.java
a est manquant. [JavadocPackage]
Vérification terminée.
Checkstyle se termine par 1 erreurs.
kkougblenou@KPLFW-COM-LKKOU MINGW64 /c:/is/activite2/src/main/java
$ |
```

Fig.01 Capture d'écran du rapport de checkstyle de la classe **Calcul.java**. L'erreur remarqué est dû au fait que notre classe ne dispose pas de fichier Javadoc utilisé pour les commentaires.

```

import static org.junit.Assert.assertEquals;
import org.junit.Test;
/** Tests unitaire pour la classe Calcul.
 *
 * @author koffi kougblenou
 */
public final class CalculTest {
    /** Déclaration de variable static. */
    static final int M = 3;
    /** Déclaration de variable static. */
    static final int Z = 8;
    /** Déclaration de variable static. */
    static final int S = 5;
    /** Déclaration de variable static. */
    static final int A = 2;
    /** Déclaration de variable static. */
    static final int B = 4;
    /** Test du constructeur . */
    @Test
    public void testConstructeur() {
        new calcul();
    }
    /** Test de la methode qui fait la somme . */
    @Test
    public void testSomme() {
        assertEquals(S, Calcul.somme(A, M));
    }
    /** Test de la methode qui fait la division . */
    @Test
    public void testDivision() {
        assertEquals(B, Calcul.division(Z, A));
    }
}

```

```

MINGW64:/c/is/activite2/src/test/java
kkougblenou@KPLFW-COM-LKKOU MINGW64 /c/is/activite2/src/test/java
$ java -jar checkstyle.jar -c sun_checks.xml CalculTest.java
Début de la vérification...
[ERROR] C:\is\activite2\src\test\java\CalculTest.java:1: Le fichier package-info.
java est manquant. [JavadocPackage]
Vérification terminée.
Checkstyle se termine par 1 erreurs.

kkougblenou@KPLFW-COM-LKKOU MINGW64 /c/is/activite2/src/test/java
$ |

```

Fig.02 Capture d'écran du rapport de checkstyle de la classe CalculTest.java. L'erreur remarqué est dû au fait que notre classe ne dispose pas de fichier Javadoc utilisé pour les commentaires.

2- Veuillez trouver ci-dessous le rapport de couverture de code :

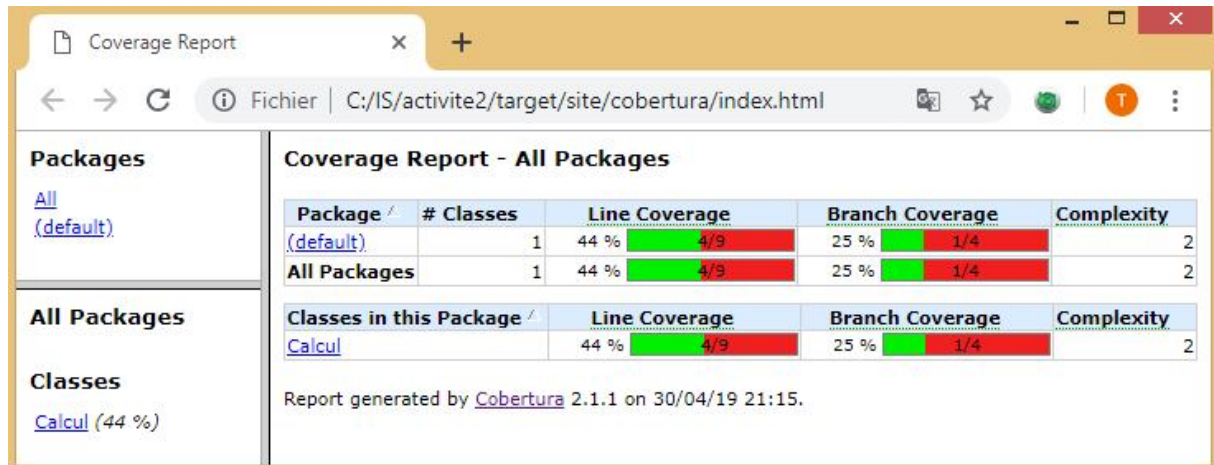


Fig.03 Capture d'écran du rapport de couverture code de la classe Calcul.java

- Capture d'écran du rapport des tests unitaires de la classe CalculTest.java sur la classe Calcul.java.

```
c:\IS\activite2\src\test\java>javac -cp junit.jar Calcul.java CalculTest.java

c:\IS\activite2\src\test\java>java -cp junit.jar;hamcrest-core.jar;. org.junit.runner.JUnitCore CalculTest
JUnit version 4.11
...
Time: 0,026

OK (3 tests)

c:\IS\activite2\src\test\java>
```

Fig.04 Capture d'écran du rapport de test unitaire de la Classe CalculTest.java sur la classe Calcul.java

3- Notre classe Calcul.java compte 9 lignes de codes exécutables et 4 méthodes et 2 conditions :

- En couverture linéaire, seules 4 lignes de codes ont été exécutés. En effet le constructeur et la méthode maFonction() ne font pas parti des tests unitaires et leurs instructions n'ont pas été testés. La proportion de ligne de codes testée est donc 4/9 soit 44%.
- En couverture conditionnelle nous avons 4 chemins logiques possibles cependant seul le else de la méthode division est testé soit une couverture de $1/4 = 25\%$.

- 4- Notre classe CalculTest.java a été réécrit comme suit. Nous avons testé ici le constructeur par défaut de la classe Calcul.java et le constructeur déclaré private a été mis en commentaire.

```
import static org.junit.Assert.*;
import org.junit.Test;
import org.junit.Rule;
import org.junit.rules.ExpectedException;
/** Tests unitaire pour la classe Calcul.
 *
 * @author koffi kougblenou
 */
public final class CalculTest {
    /** Déclaration de variable static. */
    static final int M = 3;
    /** Déclaration de variable static. */
    static final int Z = 8;
    /** Déclaration de variable static. */
    static final int S = 5;
    /** Déclaration de variable static. */
    static final int A = 2;
    /** Déclaration de variable static. */
    static final int B = 4;
    /** Déclaration de variable static. */
    static final int C = 0;
    /** Déclaration de variable static. */
    static final int P = 16;
    /** Test du constructeur . */
    @Test
    public void testConstructeur() {
        assertNotNull(new Calcul());
    }
    /** Test de la methode qui fait la somme . */
    @Test
    public void testSomme() {
        assertEquals(S, Calcul.somme(A, M));
    }
    /** Déclaration de l'exception. */
    @Rule
    public ExpectedException thrown = ExpectedException.none();
    /** Test de la methode division cas ou b vaut 0. */
    @Test
    public void testDivisionParZero() {
        thrown.expect(IllegalArgumentException.class);
        thrown.expectMessage("b ne doit pas etre 0");
        Calcul.division(Z, C);
    }
    /** Test de la methode la division cas ou b différent de 0. */
    @Test
    public void testDivision() {
        assertEquals(B, Calcul.division(Z, A));
    }
    /** Test de la méthode maFonction. */
    @Test
    public void testMaFonction() {
        assertEquals(C, Calcul.maFonction(Z, P));
        assertEquals(B, Calcul.maFonction(Z, B));
    }
}
```

Après exécution du test nous avons obtenu un rapport de couverture en ligne de code exécuté et couverture conditionnelle de 100%.

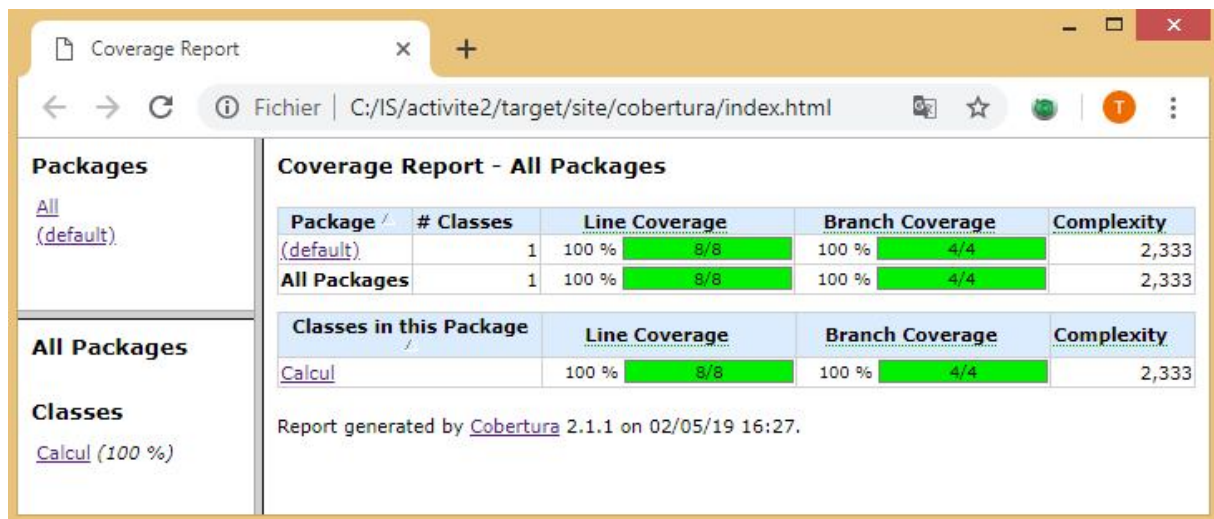


Fig.05 Capture d'écran du rapport de couverture code de la classe Calcul.java après amélioration de la classe CalculTest.java.

EXERCICE 2 : Compilation assistée

1- Pour cet exercice, l'arborescence du dossier contenant les fichiers se présente comme suit :

- sudoku\pom.xml
- sudoku\src
- sudoku\src\lib
- sudoku\src\main
- sudoku\src\main\java
- sudoku\src\test
- sudoku\src\test\java

2- Nous avons construit le fichier pom.xml en incluant les plugins et dépendances vers :

- junit:junit:4.12
- org.hamcrest: hamcrest-all:1.3
- net.sourceforge.cobertura: cobertura:2.1.1
- org.apache.maven.surefire:surefire-junit47:2.15
- org.apache.maven.plugins: maven-checkstyle-plugin:3.0.0
- org.apache.maven.plugins: maven-javadoc-plugin:3.1.0
- org.apache.maven.plugins: maven-surefire-plugin:2.15
- org.apache.maven.plugins: maven-compiler-plugin:1.6

3- Veuillez trouver ci-dessous le résultat de l'exécution de la commande complète.

```
kkougblenou@KPLFW-COM-LKKOU MINGW64 /c/IS/active2
$ mvn clean compile test checkstyle:checkstyle cobertura:cobertura javadoc:javadoc
javadoc:aggregate
[INFO] Scanning for projects...
[INFO] -----< default:mon-active2 >-----
[INFO] Building Active2 1.1
[INFO] -----[ jar ]-----
[INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ mon-active2 ---
[INFO] Deleting C:\IS\active2\target
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ mon-active2
---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\IS\active2\src\main\resources
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ mon-active2 ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to C:\IS\active2\target\classes
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ mon-active2
---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\IS\active2\src\main\resources
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ mon-active2 ---
[INFO] Nothing to compile - all classes are up to date
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ mon-
active2 ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\IS\active2\src\test\resources
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ mon-
active2 ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to C:\IS\active2\target\test-classes
[INFO] --- maven-surefire-plugin:2.15:test (default-test) @ mon-active2 ---
[INFO] Surefire report directory: C:\IS\active2\target\surefire-reports
```

T E S T S

Running Calcul Test
Tests run: 5, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.391 sec - in
Calcul Test

Results :

Tests run: 5, Failures: 0, Errors: 0, Skipped: 0

```
[INFO]
[INFO] --- maven-checkstyle-plugin:3.0.0:checkstyle (default-cli) @ mon-activite --
-
[INFO] There are 2 errors reported by Checkstyle 6.18 with sun_checks.xml ruleset.
[WARNING] Unable to locate Source XRef to link to - DISABLED
[INFO] >>> cobertura-maven-plugin:2.7:cobertura (default-cli) > [cobertura]test @
mon-activite >>>
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ mon-activite
---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\IS\activite2\src\main\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ mon-activite ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- cobertura-maven-plugin:2.7:instrument (default-cli) @ mon-activite ---
[INFO] Cobertura 2.1.1 - GNU GPL License (NO WARRANTY) - See COPYRIGHT file
[INFO] Cobertura: Saved information on 1 classes.
[INFO] Cobertura: Saved information on 1 classes.

[INFO] Instrumentation was successful.
[INFO] NOT adding cobertura ser file to attached artifacts list.
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ mon-
activite ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\IS\activite2\src\test\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ mon-
activite ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] --- maven-surefire-plugin:2.15:test (default-test) @ mon-activite ---
[INFO] Surefire report directory: C:\IS\activite2\target\surefire-reports
```

TESTS

Running Calcul Test
Tests run: 5, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 1.059 sec - in
Calcul Test
[INFO] Cobertura: Loaded information on 1 classes.
[INFO] Cobertura: Saved information on 1 classes.

Results :

Tests run: 5, Failures: 0, Errors: 0, Skipped: 0

```
[INFO]
[INFO] <<< cobertura-maven-plugin:2.7:cobertura (default-cli) < [cobertura]test @
mon-activite <<<
[INFO]
[INFO]
[INFO] --- cobertura-maven-plugin:2.7:cobertura (default-cli) @ mon-activite ---
[INFO] Cobertura 2.1.1 - GNU GPL License (NO WARRANTY) - See COPYRIGHT file
[INFO] Cobertura: Loaded information on 1 classes.
Report time: 2095ms

[INFO] Cobertura Report generation was successful.
[INFO]
[INFO] >>> maven-javadoc-plugin:3.1.0:javadoc (default-cli) > generate-sources @
mon-activite >>>
[INFO]
[INFO] <<< maven-javadoc-plugin:3.1.0:javadoc (default-cli) < generate-sources @
mon-activite <<<
[INFO]
[INFO]
[INFO] --- maven-javadoc-plugin:3.1.0:javadoc (default-cli) @ mon-activite ---
[INFO]
Loading source file C:\IS\activite2\src\main\java\Calcul.java...
Constructing Javadoc information...
Standard Doclet version 1.8.0_211
```

```

Building tree for all the packages and classes...
Generating C:\S\activite2\target\site\api docs\Calcul.html...
Generating C:\S\activite2\target\site\api docs\package-frame.html...
Generating C:\S\activite2\target\site\api docs\package-summary.html...
Generating C:\S\activite2\target\site\api docs\package-tree.html...
Generating C:\S\activite2\target\site\api docs\constant-values.html...
Generating C:\S\activite2\target\site\api docs\class-use\Calcul.html...
Generating C:\S\activite2\target\site\api docs\package-use.html...
Building index for all the packages and classes...
Generating C:\S\activite2\target\site\api docs\overview-tree.html...
Generating C:\S\activite2\target\site\api docs\index-all.html...
Generating C:\S\activite2\target\site\api docs\deprecated-list.html...
Building index for all classes...
Generating C:\S\activite2\target\site\api docs\all classes-frame.html...
Generating C:\S\activite2\target\site\api docs\all classes-noframe.html...
Generating C:\S\activite2\target\site\api docs\index.html...
Generating C:\S\activite2\target\site\api docs\help-doc.html...
[INFO] -----< default:mon-activite >-----
[INFO] Building Activite2 1.1
[INFO] -----[ jar ]-----
[INFO] >>> maven-javadoc-plugin: 3.1.0: aggregate (default-cli) > compile @ mon-
activite >>>
[INFO]
[INFO] --- maven-resources-plugin: 2.6: resources (default-resources) @ mon-activite
---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\S\activite2\src\main\resources
[INFO]
[INFO] --- maven-compiler-plugin: 3.1: compile (default-compile) @ mon-activite ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] <<< maven-javadoc-plugin: 3.1.0: aggregate (default-cli) < compile @ mon-
activite <<<
[INFO]
[INFO]
[INFO] --- maven-javadoc-plugin: 3.1.0: aggregate (default-cli) @ mon-activite ---
[INFO]
Loading source file C:\S\activite2\src\main\java\Calcul.java...
Constructing Javadoc information...
Standard Doclet version 1.8.0_211
Building tree for all the packages and classes...
Generating C:\S\activite2\target\site\api docs\Calcul.html...
Generating C:\S\activite2\target\site\api docs\package-frame.html...
Generating C:\S\activite2\target\site\api docs\package-summary.html...
Generating C:\S\activite2\target\site\api docs\package-tree.html...
Generating C:\S\activite2\target\site\api docs\constant-values.html...
Generating C:\S\activite2\target\site\api docs\class-use\Calcul.html...
Generating C:\S\activite2\target\site\api docs\package-use.html...
Building index for all the packages and classes...
Generating C:\S\activite2\target\site\api docs\overview-tree.html...
Generating C:\S\activite2\target\site\api docs\index-all.html...
Generating C:\S\activite2\target\site\api docs\deprecated-list.html...
Building index for all classes...
Generating C:\S\activite2\target\site\api docs\all classes-frame.html...
Generating C:\S\activite2\target\site\api docs\all classes-noframe.html...
Generating C:\S\activite2\target\site\api docs\index.html...
Generating C:\S\activite2\target\site\api docs\help-doc.html...
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 35.817 s
[INFO] Finished at: 2019-05-04T23:53:37Z
[INFO] -----

```

4- Le code a été déposé dans le dossier "**activite2**" du dépôt ou GIT

EXERCICE 3 :

1- Veuillez trouver ci-dessous le code des tests unitaires :

```
import static org.junit.Assert.*;
import org.junit.Test;
import org.junit.Rule;
import org.junit.rules.ExpectedException;
public class GrilleImplTest{
    static final char M = '.';
    GrilleImpl grid = new GrilleImpl();

    @Test
    public void GetDimensionTest() {
        assertEquals(9, grid.getDimension());
    }

    @Test
    public void setValueTest() {
        grid.setValue(2, 2, '9');
    }
    /** Déclaration de l'exception. */
    @Rule
    public ExpectedException thrown = ExpectedException.none();

    /** Test de la methode division cas ou b vaut 0. */
    @Test
    public void setValueTest2() {
        thrown.expect(IllegalArgumentException.class);
        thrown.expectMessage("x est hors bornes (0-8)");
        grid.setValue(9, 3, '7');
    }

    @Test
    public void setValueTest3() {
        thrown.expect(IllegalArgumentException.class);
        thrown.expectMessage("y est hors bornes (0-8)");
        grid.setValue(3, 9, '7');
    }

    @Test
    public void setValueTest4() {
        thrown.expect(IllegalArgumentException.class);
        thrown.expectMessage("value n'est pas un caractere autorise");
        grid.setValue(3, 3, '-');
    }

    @Test
    public void setValueTest5() {
        thrown.expect(IllegalArgumentException.class);
        thrown.expectMessage("la valeur est interdite aux vues des autres
valeurs de la grille");
        grid.setValue(0, 8, '9');
    }

    @Test
    public void getValueTest() {
        thrown.expect(IllegalArgumentException.class);
        thrown.expectMessage("x est hors bornes (0-8)");
        grid.getValue(9, 3);
    }
}
```

```

@Test
public void getValueTest1() {
    thrown.expect(IllegalArgumentException.class);
    thrown.expectMessage("y est hors bornes (0-8)");
    grid.getValue(3, 9);
}

@Test
public void possibleTest1() {
    thrown.expect(IllegalArgumentException.class);
    thrown.expectMessage("x est hors bornes (0-8)");
    grid.possible(9, 3, '8');
}

@Test
public void possibleTest2() {
    thrown.expect(IllegalArgumentException.class);
    thrown.expectMessage("y est hors bornes (0-8)");
    grid.possible(3, 9, '8');
}

@Test
public void possibleTest3() {
    thrown.expect(IllegalArgumentException.class);
    thrown.expectMessage("La valeur n'est pas un caractere autorise");
    grid.possible(3, 9, 'z');
}

@Test
public void possibleTest4() {
    assertTrue(grid.possible(2, 2, '7'));
}

@Test
public void possibleTest5() {
    assertFalse(grid.possible(2, 2, '4'));
}

@Test
public void checkColumnTest() {
    assertTrue(GrilleImpl.checkColumn(2, '6'));
}

@Test
public void checkColumnTest1() {
    assertFalse(GrilleImpl.checkColumn(2, '7'));
}

@Test
public void checkLineTest() {
    assertTrue(GrilleImpl.checkLine(2, '6'));
}

@Test
public void checkSquareTest() {
    assertTrue(GrilleImpl.checkSquare(2, 2, '2'));
}

@Test
public void checkSquareTest1() {
    assertFalse(GrilleImpl.checkSquare(2, 2, '7'));
}

```

```

@Test
public void completeTest() {
    grid.grille = new char[][]{
        {'9', '4', '1', '5', '3', '8', '7', '6', '2'},
        {'3', '6', '8', '9', '7', '2', '4', '5', '1'},
        {'2', '5', '7', '6', '1', '4', '8', '3', '9'},

        {'1', '2', '9', '4', '5', '6', '3', '8', '7'},
        {'6', '7', '4', '8', '9', '3', '2', '1', '5'},
        {'5', '8', '3', '1', '2', '7', '6', '9', '4'},

        {'8', '9', '6', '7', '4', '1', '5', '2', '3'},
        {'4', '1', '2', '3', '8', '5', '9', '7', '6'},
        {'7', '3', '5', '2', '6', '9', '1', '4', '8'}
    };
    assertTrue(grid.complete());
}
}

```

2- implémentation dans une classe "GrilleImpl" :

```
import java.util.Arrays;
/**
 * Implementation d'une grille .
 */
public class GrilleImpl implements Grille {
    /** Déclaration de constante. */
    private static final int NEUF = 9;
    /** Tableau de caracteres à deux dimension .*/
    public static char[][] grille = new char[][] {
        {'.', '@', '@', '@', '3', '@', '@', '6', '2'},
        {'@', '@', '@', '@', '7', '2', '@', '@', '1'},
        {'2', '@', '@', '6', '@', '@', '8', '@', '@'},
        {'@', '@', '@', '4', '5', '6', '3', '8', '7'},
        {'6', '7', '4', '@', '@', '@', '2', '@', '@'},
        {'5', '8', '@', '1', '@', '7', '6', '@', '4'},
        {'@', '@', '6', '@', '@', '1', '@', '@', '3'},
        {'4', '@', '@', '3', '8', '@', '@', '@', '@'},
        {'7', '3', '@', '@', '6', '@', '@', '@', '@'}
    };
    /**
     * @return largeur/hauteur de la grille 9 ou 16 .
     */
    public final int getDimension() {
        return grille.length;
    }
    /**
     * Test la presence de la valeur dans la colonne.
     * @param y position y dans la grille
     * @param value valeur à rechercher dans la grille
     * @return vrai si la valeur est presente sinon faux
     */
    public static boolean checkColumn(final int y, final char value) {
        for (int x = 0; x < NEUF; x++) {
            if (grille[x][y] == value) {
                return true;
            }
        }
        return false;
    }
    /**
     * Test la presence de la valeur sur la ligne.
     * @param x position x dans la grille
     * @param value valeur à rechercher dans la grille
     * @return vrai si la valeur est presente sinon faux
     */
    public static boolean checkLine(final int x, final char value) {
        for (int y = 0; y < NEUF; y++) {
            if (grille[x][y] == value) {
                return true;
            }
        }
        return false;
    }
    /**
     * Test la presence de la valeur dans le carré.
     * @param x position x dans la grille
     * @param y position y dans la grille
     * @param value valeur à rechercher dans la grille
     * @return vrai si la valeur est presente sinon faux
     */
    public static boolean
    checkSquare(final int x, final int y, final char value) {
        int dimension = grille.length;
        int leftpoint = 0;
```

```

    int toppoint = 0;
    final int q = 3;
    final int dim = 9;
    if (dimension == dim) {
        leftpoint = q * (y / q);
        toppoint = q * (x / q);
    }
    for (int c = leftpoint; c < leftpoint + q; c++) {
        for (int l = toppoint; l < toppoint + q; l++) {
            if (grille[l][c] == value) {
                return true;
            }
        }
    }
    return false;
}

/**
 * Affecte une valeur dans la grille.
 * @param x position x dans la grille
 * @param y position y dans la grille
 * @param value valeur a mettre dans la case
 * @throws IllegalArgumentException si x ou y sont hors bornes (0-8)
 * @throws IllegalArgumentException si la valeur est interdite aux vues des
 * autres valeurs de la grille
 * @throws IllegalArgumentException si value n'est pas un caractere
autorise
 * ('1', ..., '9')
 */
public final void setValue(final int x, final int y, final char value) {
    int dimension = grille.length;
    boolean cont = false;
    cont = Arrays.toString(POSSIBLE).contains(Character.toString(value));
    if (cont == true) {
        //verification
        if ((x >= NEUF) || (x < 0)) {
            throw new IllegalArgumentException("x est hors bornes (0-8)");
        } else if ((y >= NEUF) || (y < 0)) {
            throw new IllegalArgumentException("y est hors bornes (0-8)");
        } else if ((!checkColumn(y, value)) && (!checkLine(x, value))
            && (!checkSquare(x, y, value)) && !(grille[x][y] == EMPTY)) {
            throw new IllegalArgumentException("la valeur est interdite "
                + "aux vues des autres valeurs de la grille");
        } else {
            grille[x][y] = value;
        }
    } else {
        throw new IllegalArgumentException("La value n'est "
            + " pas un caractere autorise");
    }
}

/**
 * Recupere une valeur de la grille.
 * @param x position x dans la grille
 * @param y position y dans la grille
 * @return valeur dans la case x,y
 * @throws IllegalArgumentException si x ou y sont hors bornes (0-8)
 */
public final char getValue(final int x, final int y) {
    if ((x >= NEUF) || (x < 0)) {
        throw new IllegalArgumentException("x est hors bornes (0-8)");
    } else if ((y >= NEUF) || (y < 0)) {
        throw new IllegalArgumentException("y est hors bornes (0-8)");
    }
    return grille[x][y];
}

```

```

/**
 * Test si une grille est terminee.
 * @return true si la grille est complete
 */
public final boolean complete() {
    int dimension = this.getDimension();
    boolean controle = true;
    for (int i = 0; i < dimension; i++) {
        for (int j = 0; j < dimension; j++) {
            if (grille[i][j] == EMPTY) {
                controle = false;
                continue;
            }
        }
        if (controle == false) {
            continue;
        }
    }
    return controle;
}
/**
 * Test si une valeur est possible dans la grille par rapport a ce qu'elle
 * contient deja
 * @param x position x dans la grille
 * @param y position y dans la grille
 * @param value a mettre dans la case
 * @throws IllegalArgumentException si x ou y sont hors bornes (0-8)
 * @throws IllegalArgumentException si value n'est pas un caractere
autorise
 * ('1', ..., '9', ...)
 * @return true si c'est une valeur possible
 */
public final boolean possible(final int x, final int y, final char value)
{
    boolean cont = false;
    cont = Arrays.toString(POSSIBLE).contains(Character.toString(value));
    if (cont == true) {
        //verification
        if ((x >= NEUF) || (x < 0)) {
            throw new IllegalArgumentException("x est hors bornes (0-8)");
        }
        if ((y >= NEUF) || (y < 0)) {
            throw new IllegalArgumentException("y est hors bornes (0-8)");
        }
        if ((!checkColumn(y, value)) && (!checkLine(x, value))
            && (!checkSquare(x, y, value)) && (grille[x][y] ==
EMPTY)) {
            return true;
        } else {
            return false;
        }
    } else {
        throw new IllegalArgumentException("La value n'est pas "
+ "un caractere autorise");
    }
}
}

```

3 – le résultat des tests unitaires, la couverture de code sur l'exécution des tests unitaires, le rapport checkstyle :

- le résultat des tests unitaires

Surefire Report

Summary

[\[Summary\]](#) [\[Package List\]](#) [\[Test Cases\]](#)

Tests	Errors	Failures	Skipped	Success Rate	Time
19	0	0	0	100%	0,025

Note: failures are anticipated and checked for with assertions while errors are unanticipated.

Package List

[\[Summary\]](#) [\[Package List\]](#) [\[Test Cases\]](#)

Package	Tests	Errors	Failures	Skipped	Success Rate	Time
	19	0	0	0	100%	0,025

Note: package statistics are not computed recursively, they only sum up all of its testsuites numbers.

Class	Tests	Errors	Failures	Skipped	Success Rate	Time
GrilleImplTest	19	0	0	0	100%	0,025

Fig.08 Capture d'écran du rapport des tests unitaires de la classe GrilleImplTest.java sur GrilleImpl.java

- la couverture de code sur l'exécution des tests unitaires

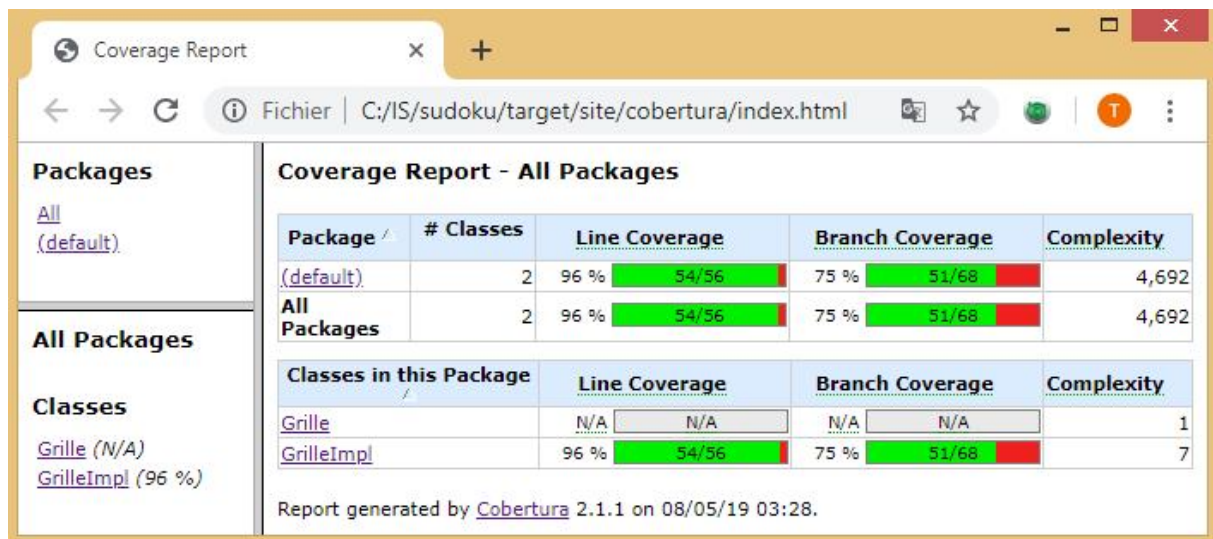
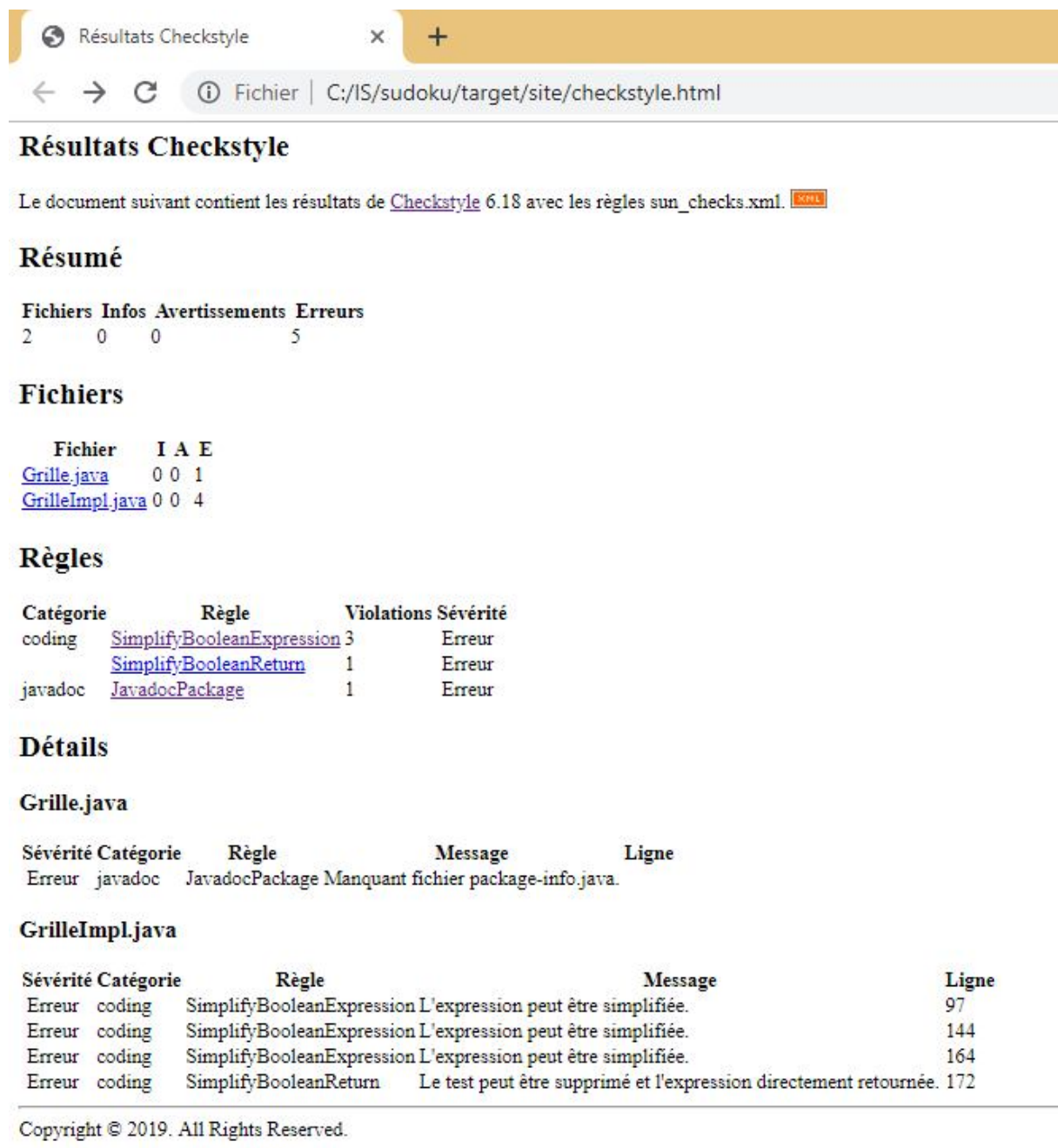


Fig.08 Capture d'écran du rapport de couverture de code de la classe GrilleImpl.java

- le rapport checkstyle



Résultats Checkstyle

Le document suivant contient les résultats de [Checkstyle 6.18](#) avec les règles [sun_checks.xml](#).

Résumé

Fichiers	Infos	Avertissements	Erreurs
2	0	0	5

Fichiers

Fichier	I	A	E
Grille.java	0	0	1
GrilleImpl.java	0	0	4

Règles

Catégorie	Règle	Violations	Sévérité
coding	SimplifyBooleanExpression	3	Erreur
	SimplifyBooleanReturn	1	Erreur
javadoc	JavadocPackage	1	Erreur

Détails

Grille.java

Sévérité	Catégorie	Règle	Message	Ligne
Erreur	javadoc	JavadocPackage	Manquant fichier package-info.java.	

GrilleImpl.java

Sévérité	Catégorie	Règle	Message	Ligne
Erreur	coding	SimplifyBooleanExpression	L'expression peut être simplifiée.	97
Erreur	coding	SimplifyBooleanExpression	L'expression peut être simplifiée.	144
Erreur	coding	SimplifyBooleanExpression	L'expression peut être simplifiée.	164
Erreur	coding	SimplifyBooleanReturn	Le test peut être supprimé et l'expression directement retournée.	172

Copyright © 2019. All Rights Reserved.

Fig.09 Capture d'écran du rapport de couverture de code des classes Grille.java et GrilleImpl.java