

Lab #02 - Rapport de tests

**Techniques de l'informatique voie de spécialisation en informatique
de gestion, profil programmation nouveaux médias (420.A0)**

Contrôle de la qualité I (420-PA4-AG)

Version : 1.0.0

Historique des révisions

Révision	Date	Description	Auteur
1.0.0	2018-09-11	Version initiale.	Christian Labelle

Table des matières

1	Introduction	1
2	Analyse de la classe à tester	1
3	Environnement de test	2
4	Description des cas de tests	3
4.1	Test #1: Élément central du tableau.....	3
4.2	Test #2: Élément existant à droite de l'élément central.....	4
4.3	Test #3: Élément existant à gauche de l'élément central.....	5
4.4	Test #4: Élément inexistant à droite de l'élément central	6
4.5	Test #5: Élément inexistant à gauche de l'élément central	7
4.6	Test #6: Tableau non-initialisé	8
5	Résultats des tests avant correctifs	9
5.1	Test #1: Élément central du tableau.....	9
5.1.1	Journal de test	9
5.2	Test #2: Élément existant à droite de l'élément central.....	10
5.2.1	Journal de test	10
5.3	Test #3: Élément existant à gauche de l'élément central.....	12
5.3.1	Journal de test	12
5.4	Test #4: Élément inexistant à droite de l'élément central	15
5.4.1	Journal de test	15
5.5	Test #5: Élément inexistant à gauche de l'élément central	16
5.5.1	Journal de test	16
5.6	Test #6: Tableau non-initialisé	17
5.6.1	Journal de test	17
6	Description des correctifs à apporter	18
7	Résultats des tests après correctifs	20
7.1	Test #1: Élément central du tableau.....	20
7.1.1	Journal de test	20
7.2	Test #2: Élément existant à droite de l'élément central.....	21
7.2.1	Journal de test	21
7.3	Test #3: Élément existant à gauche de l'élément central.....	22
7.3.1	Journal de test	22
7.4	Test #4: Élément inexistant à droite de l'élément central	23
7.4.1	Journal de test	23
7.5	Test #5: Élément inexistant à gauche de l'élément central	24
7.5.1	Journal de test	24
7.6	Test #6: Tableau non-initialisé	25
7.6.1	Journal de test	25

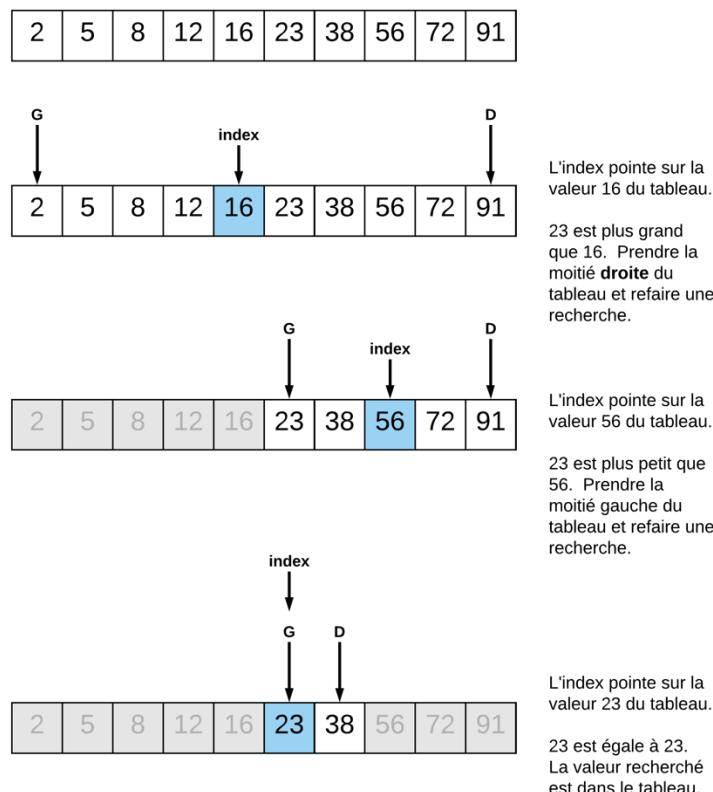
1 INTRODUCTION

Ce document décrit le fonctionnement de la classe *Java* « *RechercheBinaire* », l'environnement utilisé pour effectuer les tests, les cas de test exécutés, une description des correctifs apportés ainsi que les résultats des tests avant et après l'implémentation des correctifs. Il consigne toute l'information liée au traitement des anomalies de façon minutieuse incluant les hypothèses pertinentes sur la nature du problème, le choix des tests visant à vérifier les hypothèses et les jeux d'essai appropriés. Il montre également que les tests unitaires sont fonctionnels, complets et pertinents.

2 ANALYSE DE LA CLASSE À TESTER

La classe *Java* « *RechercheBinaire* » implémente un algorithme de recherche de valeur dans un tableau de nombre entier trié en ordre croissant. Au lieu de rechercher le tableau séquentiellement, la méthode de recherche commence par examiner l'élément central. Si la valeur est celle que nous recherchons, la recherche se termine avec succès. Si ce n'est pas la bonne valeur, l'implémentation utilise la nature ordonnée du tableau pour éliminer la moitié des éléments. Ainsi, si l'élément recherché est inférieur à l'élément central, toute la moitié supérieure (droite) du tableau ainsi que la valeur centrale peuvent être éliminées. De même, si l'élément est supérieur à l'élément central, toute la moitié inférieure (gauche) du tableau ainsi que la valeur centrale peuvent être éliminées. Le processus est répété avec les éléments restants jusqu'à ce que l'élément ait été trouvé ou que tous les éléments aient été éliminés. Le diagramme ci-dessous montre comment trouver rapidement la valeur 23 dans un tableau d'entiers préalablement trié en ordre croissant :

Recherche de 23 dans un tableau de 10 éléments



3 ENVIRONNEMENT DE TEST

Matériel	
Modèle	MacBook Pro
Processeur	Intel Core i7
Vitesse du processeur	2.9 GHz
Nombre de processeurs	1
Nombre de « Cores »	4
L2 Cache (par « Core »)	256 KB
L3 Cache	8 MB
Mémoire	16 GB
Version Boot ROM	MBP143.0178.B00
Disque dur	
Type	SSD
Capacité	499.96 GB
Espace disponible	365.34 GB
Logiciels	
Système d'exploitation	macOS 10.13.6 (17G65)
Version « Kernel »	Darwin 17.7.0
JDK	Java version « 1.8.0_181 » Java(TM) SE Runtime Environment (build 1.8.0_181-b13) Java HotSpot(TM) 64-Bit Server VM (build 25.181-b13, mixed mode)
Eclipse	Eclipse IDE for Java Developers Version: Photon Release (4.8.0) Build id: 20180619-1200 OS: Mac OS X, v.10.13.6, x86_64 / cocoa Java version: 1.8.0_181
JUnit Testing Framework	4.12.0.v201504281640

4 DESCRIPTION DES CAS DE TESTS

4.1 TEST #1: ÉLÉMENT CENTRAL DU TABLEAU

Description	Le test #1 exécute la méthode « <code>rechercher</code> » de la classe Java « <code>RechercheBinaire</code> » pour trouver l'élément positionné au centre d'un tableau d'entiers.		
Pré-conditions	1. Le tableau dans lequel la recherche de l'élément est effectuée est initialisé avec des valeurs entières. 2. Le tableau dans lequel la recherche de l'élément est effectuée est trié en ordre croissant de valeurs.		
Étape	Action	Résultat attendu	Commentaire
1	Initialisation du tableau		Un tableau d'entier est initialisé avec les valeurs { 1, 2, 3, 4, 5, 6, 7, 8 }.
2	Initialisation de la valeur recherchée		La valeur recherchée est initialisée à 4.
3	Comparaison #1	La valeur recherchée est égale à la valeur trouvée : $4 == 4$	Bornes de recherche : <ul style="list-style-type: none">• gauche = 0• droit = 7 Valeur élément central : 4
Post-condition	La valeur 4 est trouvée dans le tableau d'entiers.		

4.2 TEST #2: ÉLÉMENT EXISTANT À DROITE DE L'ÉLÉMENT CENTRAL

Description	Le test #2 exécute la méthode « rechercher » de la classe Java « RechercheBinaire » pour trouver un élément positionné à droite de l'élément central du tableau d'entiers.		
Pré-conditions	1. Le tableau dans lequel la recherche de l'élément est effectuée est initialisé avec des valeurs entières. 2. Le tableau dans lequel la recherche de l'élément est effectuée est trié en ordre croissant de valeurs.		
Étape	Action	Résultat attendu	Commentaire
1	Initialisation du tableau		Un tableau d'entier est initialisé avec les valeurs { 1, 2, 3, 4, 5, 6, 7, 8 }.
2	Initialisation de la valeur recherchée		La valeur recherchée est initialisée à 5.
3	Comparaison #1	La valeur recherchée est supérieure à la valeur trouvée : $5 > 4$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 0 • droit = 7 Valeur élément central : 4
4	Comparaison #2	La valeur recherchée est inférieure à la valeur trouvée : $5 < 6$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 4 • droit = 7 Valeur élément central : 6
5	Comparaison #3	La valeur recherchée est égale à la valeur trouvée : $5 == 5$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 4 • droit = 4 Valeur élément central : 5
Post-condition	La valeur 5 est trouvée dans le tableau d'entiers.		

4.3 TEST #3: ÉLÉMENT EXISTANT À GAUCHE DE L'ÉLÉMENT CENTRAL

Description	Le test #3 exécute la méthode « rechercher » de la classe Java « RechercheBinaire » pour trouver un élément positionné à gauche de l'élément central du tableau d'entiers.		
Pré-conditions	1. Le tableau dans lequel la recherche de l'élément est effectuée est initialisé avec des valeurs entières. 2. Le tableau dans lequel la recherche de l'élément est effectuée est trié en ordre croissant de valeurs.		
Étape	Action	Résultat attendu	Commentaire
1	Initialisation du tableau		Un tableau d'entier est initialisé avec les valeurs { 1, 2, 3, 4, 5, 6, 7, 8 }.
2	Initialisation de la valeur recherchée		La valeur recherchée est initialisée à 1.
3	Comparaison #1	La valeur recherchée est supérieure à la valeur trouvée : $1 < 4$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 0 • droit = 7 Valeur élément central : 4
4	Comparaison #2	La valeur recherchée est inférieure à la valeur trouvée : $1 < 2$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 0 • droit = 2 Valeur élément central : 2
5	Comparaison #3	La valeur recherchée est égale à la valeur trouvée : $1 == 1$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 0 • droit = 0 Valeur élément central : 1
Post-condition	La valeur 1 est trouvée dans le tableau d'entiers.		

4.4 TEST #4: ÉLÉMENT INEXISTANT À DROITE DE L'ÉLÉMENT CENTRAL

Description	Le test #4 exécute la méthode « rechercher » de la classe Java « RechercheBinaire » pour trouver un élément positionné à droite de l'élément central du tableau d'entiers.		
Pré-conditions	1. Le tableau dans lequel la recherche de l'élément est effectuée est initialisé avec des valeurs entières. 2. Le tableau dans lequel la recherche de l'élément est effectuée est trié en ordre croissant de valeurs.		
Étape	Action	Résultat attendu	Commentaire
1	Initialisation du tableau		Un tableau d'entier est initialisé avec les valeurs { 1, 2, 3, 4, 5, 6, 7, 8 }.
2	Initialisation de la valeur recherchée		La valeur recherchée est initialisée à 10.
3	Comparaison #1	La valeur recherchée est supérieure à la valeur trouvée : $10 > 4$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 0 • droit = 7 Valeur élément central : 4
4	Comparaison #2	La valeur recherchée est supérieure à la valeur trouvée : $10 > 6$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 4 • droit = 7 Valeur élément central : 6
5	Comparaison #3	La valeur recherchée est supérieure à la valeur trouvée : $10 > 7$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 6 • droit = 7 Valeur élément central : 7
6	Comparaison #4	La valeur recherchée est supérieure à la valeur trouvée : $10 > 8$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 7 • droit = 7 Valeur élément central : 8
7	Conclusion	Tous les éléments du tableau ont été éliminés. La recherche est terminée.	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 8 • droit = 7
Post-condition	La valeur 10 n'est pas trouvée dans le tableau d'entiers.		

4.5 TEST #5: ÉLÉMENT INEXISTANT À GAUCHE DE L'ÉLÉMENT CENTRAL

Description	Le test #5 exécute la méthode « rechercher » de la classe Java « RechercheBinaire » pour trouver un élément positionné à gauche de l'élément central du tableau d'entiers.		
Pré-conditions	1. Le tableau dans lequel la recherche de l'élément est effectuée est initialisé avec des valeurs entières. 2. Le tableau dans lequel la recherche de l'élément est effectuée est trié en ordre croissant de valeurs.		
Étape	Action	Résultat attendu	Commentaire
1	Initialisation du tableau		Un tableau d'entier est initialisé avec les valeurs { 1, 2, 3, 4, 5, 6, 7, 8 }.
2	Initialisation de la valeur recherchée		La valeur recherchée est initialisée à 0.
3	Comparaison #1	La valeur recherchée est inférieure à la valeur trouvée : $0 < 4$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 0 • droit = 7 Valeur élément central : 4
4	Comparaison #2	La valeur recherchée est inférieure à la valeur trouvée : $0 < 2$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 0 • droit = 2 Valeur élément central : 2
5	Comparaison #3	La valeur recherchée est inférieure à la valeur trouvée : $0 < 1$	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 0 • droit = 0 Valeur élément central : 1
6	Conclusion	Tous les éléments du tableau ont été éliminés. La recherche est terminée.	Bornes de recherche : <ul style="list-style-type: none"> • gauche = 0 • droit = -1
Post-condition	La valeur 0 n'est pas trouvée dans le tableau d'entiers.		

4.6 TEST #6: TABLEAU NON-INITIALISÉ

Description	Le test #6 exécute la méthode « rechercher » de la classe Java « RechercheBinaire » avec un tableau d'entiers non initialisé.		
Pré-conditions	Le tableau dans lequel la recherche de l'élément est effectuée n'est pas initialisé.		
Étape	Action	Résultat attendu	Commentaire
1	Initialisation de la valeur recherchée		La valeur recherchée est initialisée à 0.
2	Démarrage de la recherche sans initialiser le tableau d'entiers.	java.lang.NullPointerException	tableau = null
Post-condition	Une exception « java.lang.NullPointerException » est lancée.		

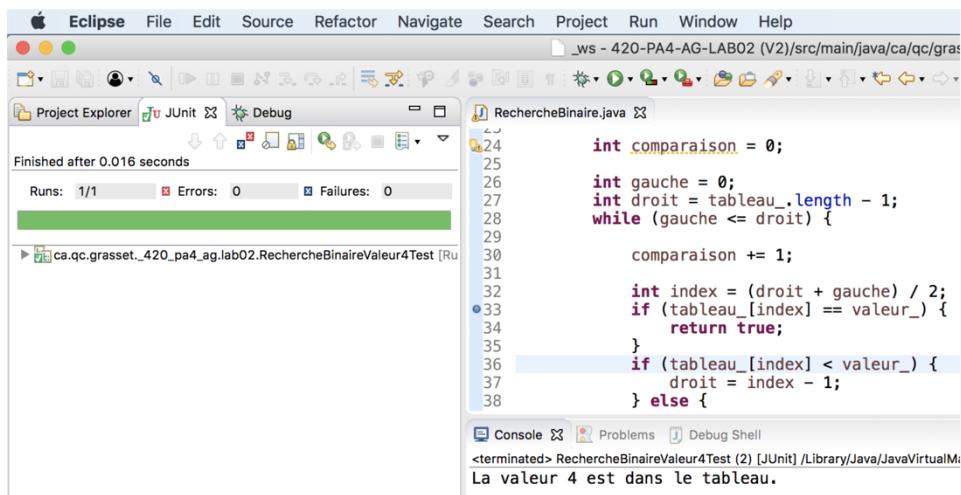
5 RÉSULTATS DES TESTS AVANT CORRECTIFS

5.1 TEST #1: ÉLÉMENT CENTRAL DU TABLEAU

Succès

La valeur 4 est trouvée dans le tableau d'entiers.

5.1.1 JOURNAL DE TEST



The screenshot shows the Eclipse IDE interface. The top menu bar includes Eclipse, File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The title bar indicates the current workspace is '_ws - 420-PA4-AG-LAB02 (V2)/src/main/java/ca/qc/grasset/_'. The left side features the Project Explorer view, which shows a folder named 'RechercheBinaire.java' and displays the message 'Finished after 0.016 seconds'. Below it, the status bar shows 'Runs: 1/1 Errors: 0 Failures: 0'. The central area contains the code editor with the file 'RechercheBinaire.java' open. The code implements a binary search algorithm:

```
int comparaison = 0;
int gauche = 0;
int droit = tableau_.length - 1;
while (gauche <= droit) {
    comparaison += 1;
    int index = (droit + gauche) / 2;
    if (tableau_[index] == valeur_) {
        return true;
    }
    if (tableau_[index] < valeur_) {
        droit = index - 1;
    } else {
```

The right side of the interface includes the Console, Problems, and Debug Shell panes. The Console pane shows the output of the test: '<terminated> RechercheBinaireValeur4Test (2) [JUnit] /Library/Java/JavaVirtualMachine'. Below this, the text 'La valeur 4 est dans le tableau.' is displayed.

5.2 TEST #2: ÉLÉMENT EXISTANT À DROITE DE L'ÉLÉMENT CENTRAL

Échec

La valeur 5 n'est pas trouvée dans le tableau d'entiers.

Description		Résultat attendu	Résultat actuel
Comparaison #1	Bornes de recherche		
	gauche	0	0
	droit	7	7
	Valeur élément central	4	4
Comparaison #2	Bornes de recherche		
	gauche	4	0
	droit	7	2
	Valeur élément central	6	2

5.2.1 JOURNAL DE TEST

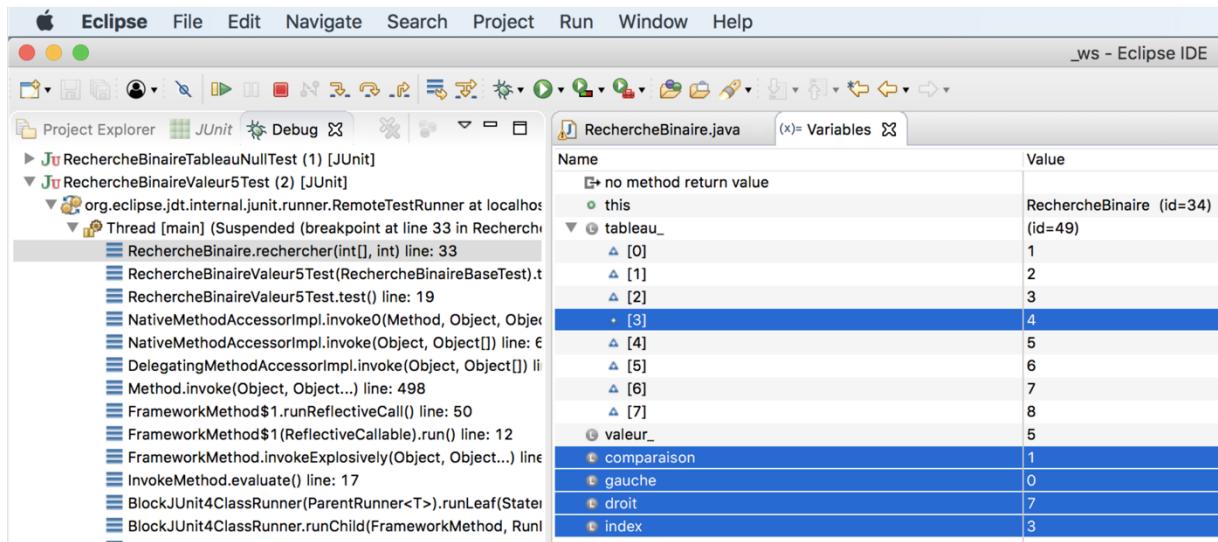
The screenshot shows the Eclipse IDE interface. The top menu bar includes Eclipse, File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The title bar indicates the current workspace is '_ws - 420-PA4-AG-LAB02 (V2)/src/main/java/ca/qc/grasset_420_pa4_ag.lab02'. The left side features the Project Explorer view, which shows a 'JUnit' entry and a 'Debug' entry. Below the Project Explorer is a progress bar indicating the test finished after 0.02 seconds. The center of the screen displays the code editor for 'RechercheBinaire.java'. The code implements a binary search algorithm. The right side shows the 'Console' view, which outputs the message 'La valeur 5 n'est pas dans le tableau.' (The value 5 is not in the array.)

```
int comparaison = 0;
int gauche = 0;
int droit = tableau_.length - 1;
while (gauche <= droit) {
    comparaison += 1;
    int index = (droit + gauche) / 2;
    if (tableau_[index] == valeur_) {
        return true;
    }
    if (tableau_[index] < valeur_) {
        droit = index - 1;
    } else {
```

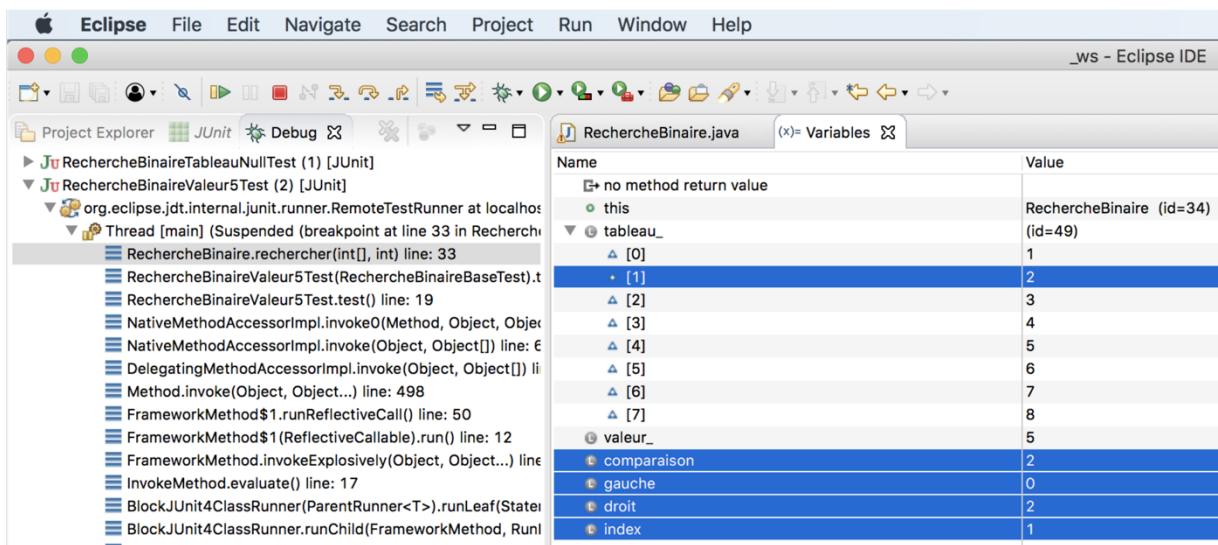
Console Problems Debug Shell

<terminated> RechercheBinaireValeur5Test (2) [JUnit] /Library/Java/JavaVirtualMachine
La valeur 5 n'est pas dans le tableau.

5.2.1.1 Comparaison #1



5.2.1.2 Comparaison #2



5.3 TEST #3: ÉLÉMENT EXISTANT À GAUCHE DE L'ÉLÉMENT CENTRAL

Échec

La valeur 1 n'est pas trouvée dans le tableau d'entiers.

Description		Résultat attendu	Résultat actuel
Comparaison #1	Bornes de recherche		
	gauche	0	0
	droit	7	7
	Valeur élément central	4	4
Comparaison #2	Bornes de recherche		
	gauche	0	4
	droit	2	7
	Valeur élément central	2	6

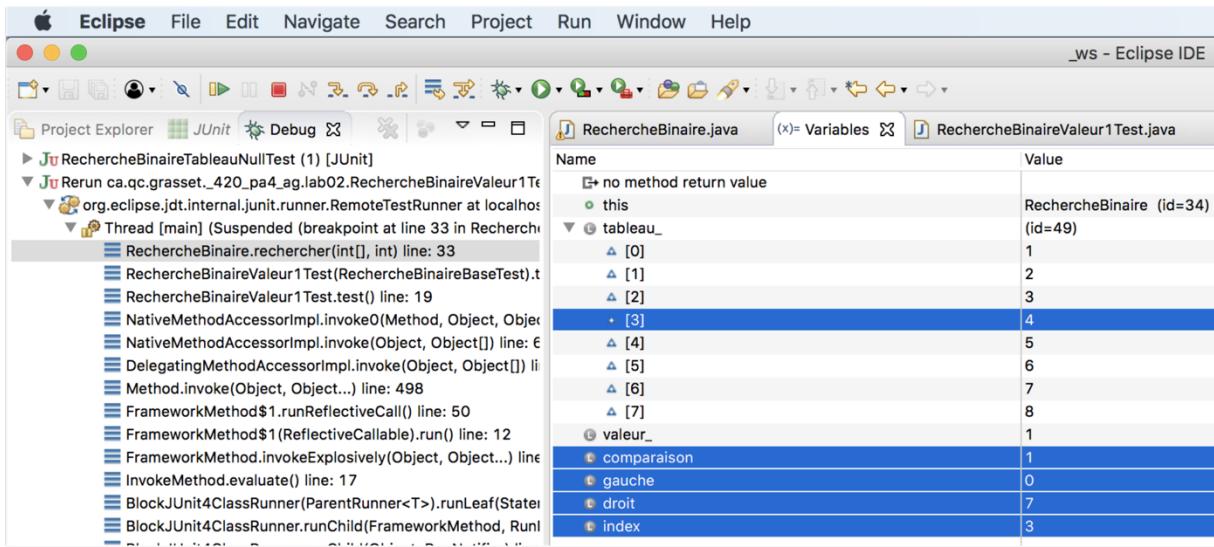
5.3.1 JOURNAL DE TEST

The screenshot shows the Eclipse IDE interface with the following details:

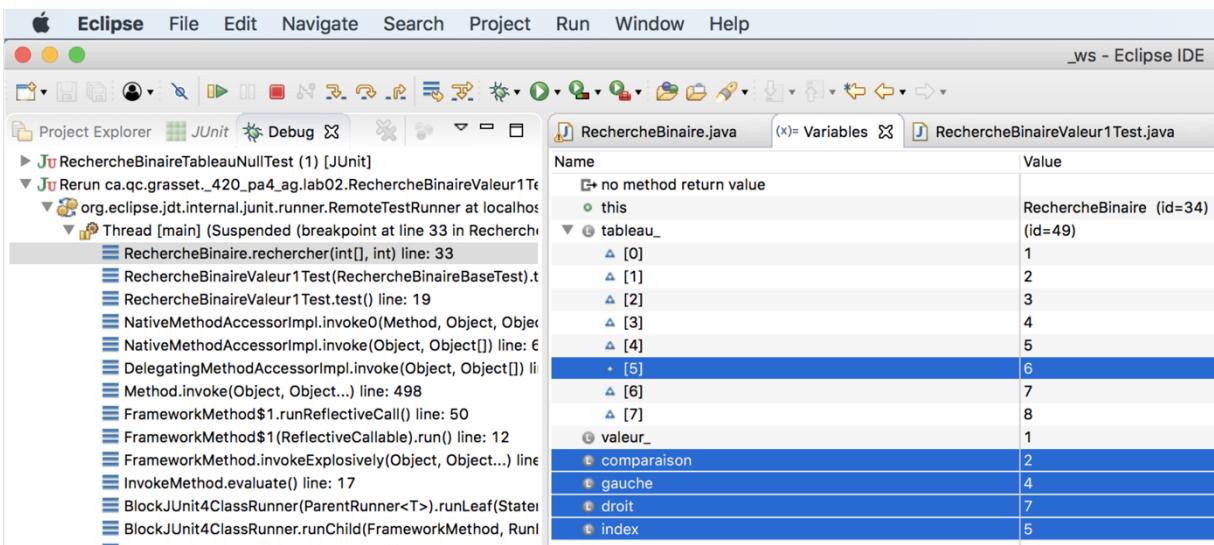
- Toolbar:** Eclipse, File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Project Explorer:** Shows a Java project structure.
- Console:** Displays the output of the test run:

```
<terminated> RechercheBinaireValeur1Test (1) [JUnit] /Library/Java/JavaVirtualMachine
La valeur 1 n'est pas dans le tableau.
```
- Code Editor:** Displays the Java code for `RechercheBinaire.java`. The failing line is highlighted at line 36: `if (tableau_[index] < valeur_) {`.

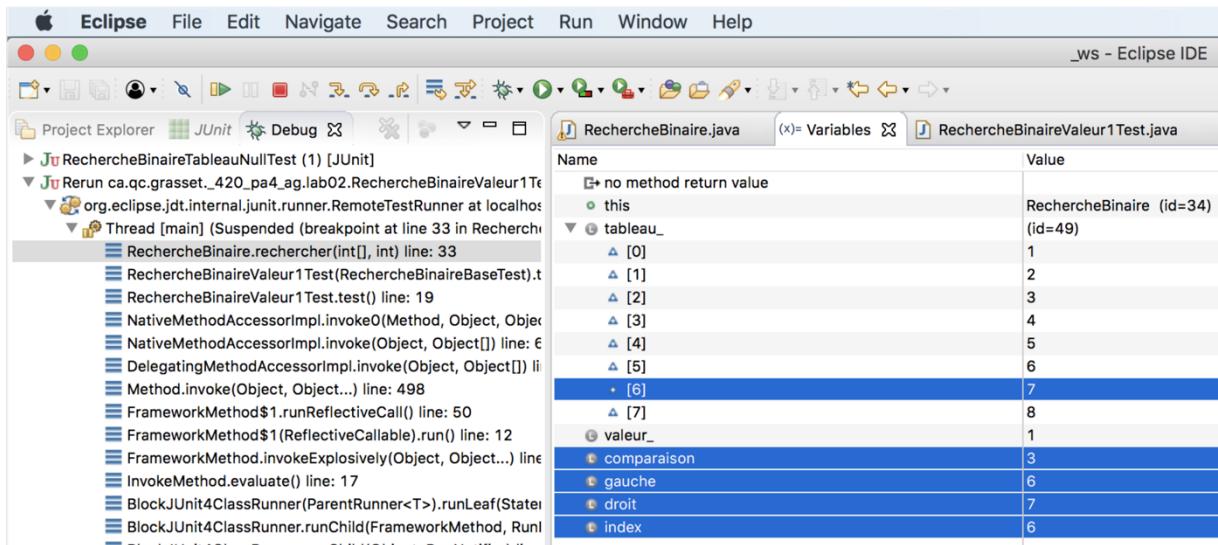
5.3.1.1 Comparaison #1



5.3.1.2 Comparaison #2



5.3.1.3 Comparaison #3

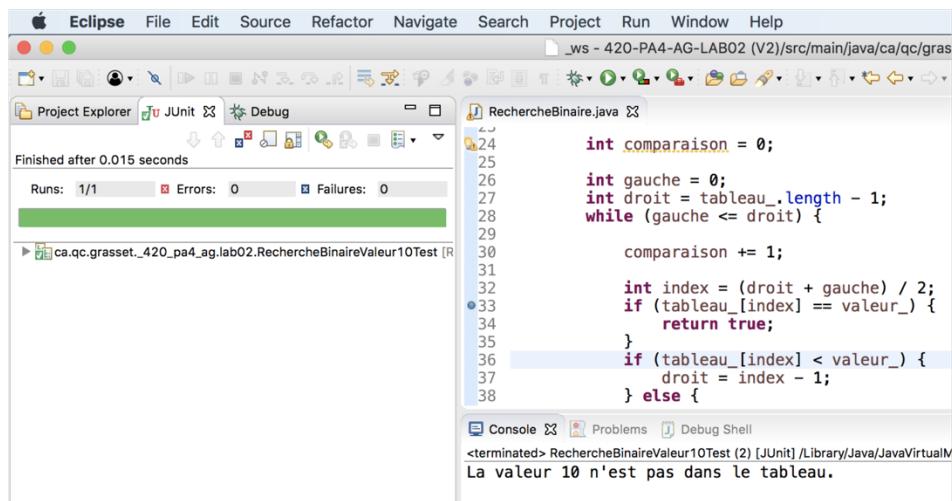


5.4 TEST #4: ÉLÉMENT INEXISTANT À DROITE DE L'ÉLÉMENT CENTRAL

Succès

La valeur 10 n'est pas trouvée dans le tableau d'entiers.

5.4.1 JOURNAL DE TEST



The screenshot shows the Eclipse IDE interface. The top menu bar includes Eclipse, File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The title bar indicates the current workspace is '_ws - 420-PA4-AG-LAB02 (V2)/src/main/java/ca/qc/gras'. The left side features a Project Explorer view with a green progress bar at the bottom labeled 'Finished after 0.015 seconds' and 'Runs: 1/1 Errors: 0 Failures: 0'. The center-right area contains a code editor with Java code for a binary search algorithm. The code is as follows:

```
int comparaison = 0;
int gauche = 0;
int droit = tableau_.length - 1;
while (gauche <= droit) {
    comparaison += 1;
    int index = (droit + gauche) / 2;
    if (tableau_[index] == valeur_) {
        return true;
    }
    if (tableau_[index] < valeur_) {
        droit = index - 1;
    } else {
```

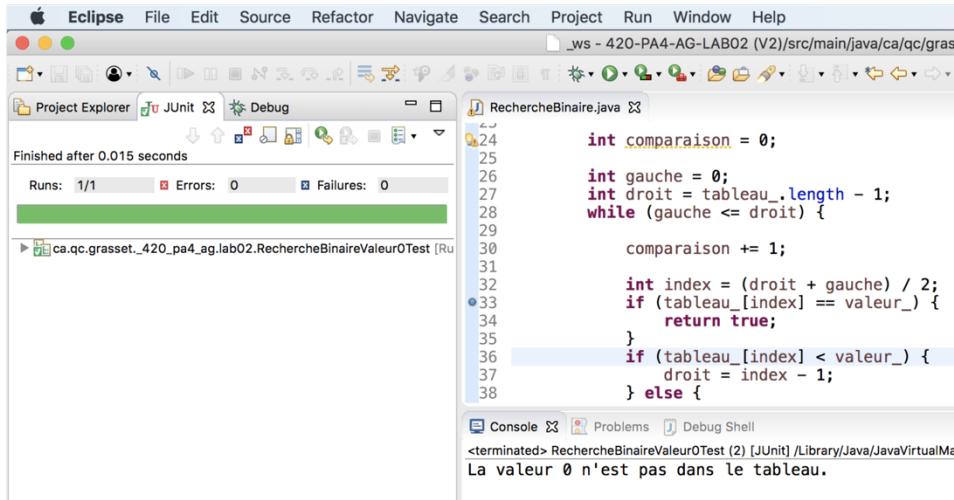
The line 'if (tableau_[index] < valeur_) {' is highlighted in blue. Below the code editor, the Console tab shows the output: '<terminated> RechercheBinaireValeur10Test (2) [JUnit] /Library/Java/JavaVirtualM'. The message 'La valeur 10 n'est pas dans le tableau.' is displayed in the console window.

5.5 TEST #5: ÉLÉMENT INEXISTANT À GAUCHE DE L'ÉLÉMENT CENTRAL

Succès

La valeur 0 n'est pas trouvée dans le tableau d'entiers.

5.5.1 JOURNAL DE TEST



The screenshot shows the Eclipse IDE interface. The top menu bar includes Eclipse, File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The title bar indicates the current workspace is '_ws - 420-PA4-AG-LAB02 (V2)/src/main/java/ca/qc/gras'. The left side features the Project Explorer, JUnit, and Debug perspectives. The JUnit perspective is active, showing a green progress bar with 'Runs: 1/1', 'Errors: 0', and 'Failures: 0'. Below the progress bar is a message: 'ca.qc.grasset._420_pa4_ag.lab02.RechercheBinaireValeur0Test [Run]'. The right side displays the code editor for 'RechercheBinaire.java' with the following content:

```
int comparaison = 0;
int gauche = 0;
int droit = tableau_.length - 1;
while (gauche <= droit) {
    comparaison += 1;
    int index = (droit + gauche) / 2;
    if (tableau_[index] == valeur_) {
        return true;
    }
    if (tableau_[index] < valeur_) {
        droit = index - 1;
    } else {
```

The console view at the bottom shows the output: '<terminated> RechercheBinaireValeur0Test (2) [JUnit] /Library/Java/JavaVirtualMachine'. A message box displays the text: 'La valeur 0 n'est pas dans le tableau.'

5.6 TEST #6: TABLEAU NON-INITIALISÉ

Succès

Une exception « java.lang.NullPointerException » est lancée.

5.6.1 JOURNAL DE TEST

The screenshot shows the Eclipse IDE interface with the following details:

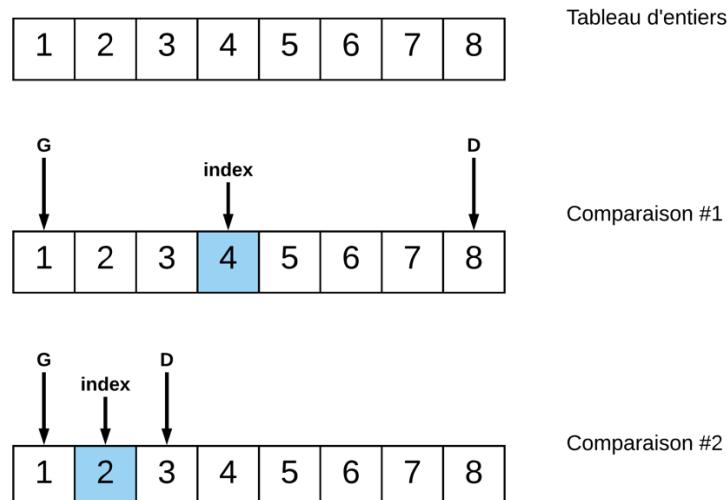
- Project Explorer:** Shows a single project named "RechercheBinaireTableauNullTest".
- Run Status:** "Runs: 1/1 Errors: 0 Failures: 0" completed after 0.017 seconds.
- Code Editor:** Displays the source code for `RechercheBinaire.java`. The code includes a search requirement and a search method. A `NullPointerException` is highlighted in red at the line `at ca.qc.grasset._420_pa4_ag.lab02.RechercheBinaire.rechercher(RechercheBinaire.java:27)`.
- Console:** Shows the command `java.lang.NullPointerException` followed by the stack trace:

```
terminated> RechercheBinaireTableauNullTest (2) [JUnit] /Library/Java/JavaVirtualMachines/dk1.8.0_181.jdk/Contents/Home/bin/java (Sep 17, 2018, 8:24:44 PM)  
java.lang.NullPointerException  
at ca.qc.grasset._420_pa4_ag.lab02.RechercheBinaire.rechercher(RechercheBinaire.java:27)  
at ca.qc.grasset._420_pa4_ag.lab02.RechercheBinaireTableauNullTest.test(RechercheBinaire)  
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)  
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
```

6 DESCRIPTION DES CORRECTIFS À APPORTER

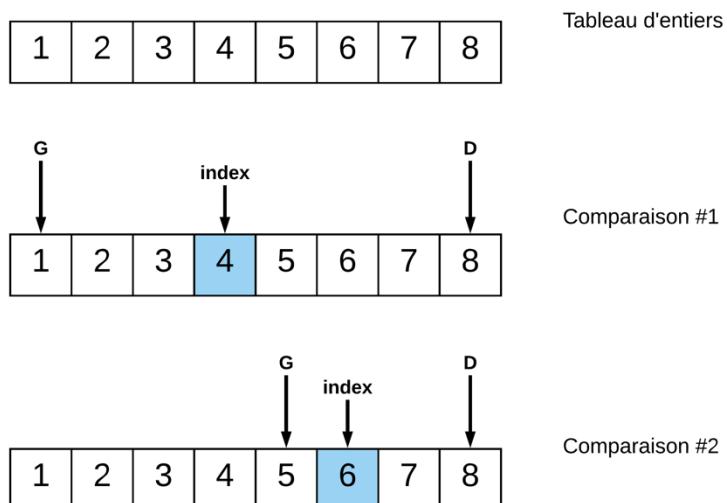
Comme le montre le diagramme ci-dessous, lors du test #2, la recherche de la valeur 5 se fait dans les éléments à gauche de l'élément central. Les bornes inférieures et supérieures ne sont pas calculées correctement pour la deuxième comparaison:

Recherche de 5 dans un tableau de 8 éléments



De même, lors du test #3, la recherche de la valeur 1 se fait dans les éléments à droite de l'élément central. Encore une fois, les bornes inférieures et supérieures ne sont pas calculées correctement pour la deuxième comparaison:

Recherche de 1 dans un tableau de 8 éléments



L'hypothèse posée pour expliquer le problème est que l'initialisation des bornes de recherche est inversée. Le correctif proposé est donc de permute la condition d'initialisation utilisée dans le code en changeant l'opérateur de comparaison :

Condition avant correctif:

```
public boolean rechercher(  
    final int[] tableau,  
    final int valeur) {  
  
    int gauche = 0;  
    int droit = tableau.length - 1;  
    while (gauche <= droit) {  
        int index = (droit + gauche) / 2;  
        if (tableau[index] == valeur) {  
            return true;  
        }  
        if (tableau[index] < valeur) {  
            droit = index - 1;  
        } else {  
            gauche = index + 1;  
        }  
    }  
    return false;  
}
```

Condition après correctif:

```
public boolean rechercher(  
    final int[] tableau,  
    final int valeur) {  
  
    int gauche = 0;  
    int droit = tableau.length - 1;  
    while (gauche <= droit) {  
        int index = (droit + gauche) / 2;  
        if (tableau[index] == valeur) {  
            return true;  
        }  
        if (tableau[index] > valeur) {  
            droit = index - 1;  
        } else {  
            gauche = index + 1;  
        }  
    }  
    return false;  
}
```

7 RÉSULTATS DES TESTS APRÈS CORRECTIFS

7.1 TEST #1: ÉLÉMENT CENTRAL DU TABLEAU

Succès

La valeur 4 est trouvée dans le tableau d'entiers.

7.1.1 JOURNAL DE TEST

The screenshot shows the Eclipse IDE interface with the following details:

- Toolbar:** Standard Eclipse toolbar with icons for file operations, search, and project navigation.
- Menu Bar:** Eclipse, File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, Help.
- Project Explorer:** Shows a Java project structure.
- JUnit View:** Displays the status of the test run: "Finished after 0.016 seconds" with 1/1 runs, 0 errors, and 0 failures.
- Code Editor:** Displays the Java code for `RechercheBinaire.java`. The code implements a binary search algorithm to find the value 4 in an array. The code is as follows:

```
int comparaison = 0;
int gauche = 0;
int droit = tableau_.length - 1;
while (gauche <= droit) {
    comparaison += 1;
    int index = (droit + gauche) / 2;
    if (tableau_[index] == valeur_) {
        return true;
    }
    if (tableau_[index] > valeur_) {
        droit = index - 1;
    } else {

```

- Console View:** Shows the output of the test: "`<terminated> RechercheBinaireValeur4Test (2) [JUnit]`" and the message "`La valeur 4 est dans le tableau.`".

7.2 TEST #2: ÉLÉMENT EXISTANT À DROITE DE L'ÉLÉMENT CENTRAL

Succès

La valeur 5 est trouvée dans le tableau d'entiers.

7.2.1 JOURNAL DE TEST

The screenshot shows the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The title bar indicates the current project is "ws - 420-PA4-AG-LAB02 (V2)/src/main/java/ca/qc/gras". The left side features a Project Explorer view with a green progress bar at the bottom labeled "Finished after 0.016 seconds" and statistics: "Runs: 1/1", "Errors: 0", and "Failures: 0". A JUnit view shows a single test named "RechercheBinaireValeur5Test" has run successfully. The main editor window displays Java code for a binary search algorithm. The console view at the bottom right shows the output: "<terminated> RechercheBinaireValeur5Test (2) [JUnit] /Library/Java/JavaVirtualMachine La valeur 5 est dans le tableau." This output corresponds to the message in the success box above.

```
int comparaison = 0;
int gauche = 0;
int droit = tableau_.length - 1;
while (gauche <= droit) {
    comparaison += 1;
    int index = (droit + gauche) / 2;
    if (tableau_[index] == valeur_) {
        return true;
    }
    if (tableau_[index] > valeur_) {
        droit = index - 1;
    } else {
```

7.3 TEST #3: ÉLÉMENT EXISTANT À GAUCHE DE L'ÉLÉMENT CENTRAL

Succès

La valeur 1 est trouvée dans le tableau d'entiers.

7.3.1 JOURNAL DE TEST

The screenshot shows the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The title bar indicates the current workspace is '_ws - 420-PA4-AG-LAB02 (V2)/src/main/java/ca/qc/gras'. The left side features the Project Explorer, JUnit, and Debug perspectives. The JUnit perspective is active, showing a green bar indicating 'Finished after 0.015 seconds' with 'Runs: 1/1', 'Errors: 0', and 'Failures: 0'. Below this, a test class 'ca.qc.grasset._420_pa4_ag.lab02.RechercheBinaireValeur1Test' is listed. The right side displays the code editor for 'RechercheBinaire.java' and the Console view. The code implements a binary search algorithm:

```
int comparaison = 0;
int gauche = 0;
int droit = tableau_.length - 1;
while (gauche <= droit) {
    comparaison += 1;
    int index = (droit + gauche) / 2;
    if (tableau_[index] == valeur_) {
        return true;
    }
    if (tableau_[index] > valeur_) {
        droit = index - 1;
    } else {
```

The Console view shows the output: '<terminated> RechercheBinaireValeur1Test (1) [JUnit] /Library/Java/JavaVirtualMachine'. Below it, the message 'La valeur 1 est dans le tableau.' is displayed.

7.4 TEST #4: ÉLÉMENT INEXISTANT À DROITE DE L'ÉLÉMENT CENTRAL

Succès

La valeur 10 n'est pas trouvée dans le tableau d'entiers.

7.4.1 JOURNAL DE TEST

The screenshot shows the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The title bar indicates the current workspace is '_ws - 420-PA4-AG-LAB02 (V2)/src/main/java/ca/qc/gras'. The left side features the Project Explorer view, which shows a JUnit icon and a Debug icon. Below it, a status bar displays 'Finished after 0.018 seconds', 'Runs: 1/1', 'Errors: 0', and 'Failures: 0'. The central area contains the code editor for 'RechercheBinaire.java' and the Console view. The code editor shows the following Java code:

```
int comparaison = 0;
int gauche = 0;
int droit = tableau_.length - 1;
while (gauche <= droit) {
    comparaison += 1;
    int index = (droit + gauche) / 2;
    if (tableau_[index] == valeur_) {
        return true;
    }
    if (tableau_[index] > valeur_) {
        droit = index - 1;
    } else {
```

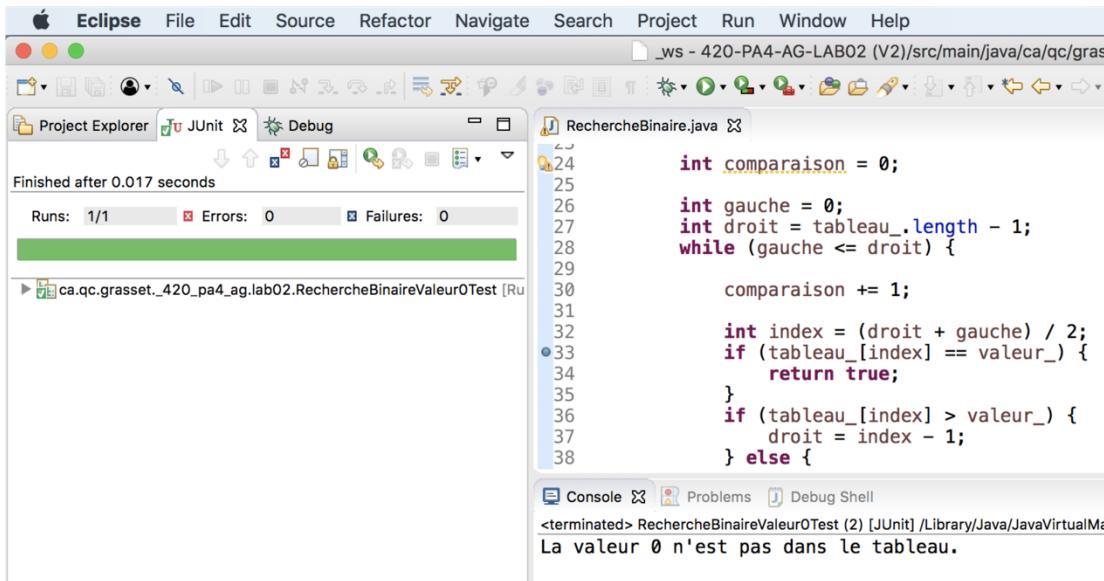
The Console view at the bottom shows the output of the test: '<terminated> RechercheBinaireValeur10Test (2) [JUnit] /Library/Java/JavaVirtualM La valeur 10 n'est pas dans le tableau.'

7.5 TEST #5: ÉLÉMENT INEXISTANT À GAUCHE DE L'ÉLÉMENT CENTRAL

Succès

La valeur 0 n'est pas trouvée dans le tableau d'entiers.

7.5.1 JOURNAL DE TEST



The screenshot shows the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The title bar indicates the current workspace is 'ws - 420-PA4-AG-LABO2 (V2)/src/main/java/ca/qc/grasset'. The left side features the Project Explorer, JUnit, and Debug perspectives. The center-right area displays the code editor for 'RechercheBinaire.java' with the following content:

```
int comparaison = 0;
int gauche = 0;
int droit = tableau_.length - 1;
while (gauche <= droit) {
    comparaison += 1;
    int index = (droit + gauche) / 2;
    if (tableau_[index] == valeur_) {
        return true;
    }
    if (tableau_[index] > valeur_) {
        droit = index - 1;
    } else {
```

The bottom right corner of the code editor shows the output: 'La valeur 0 n'est pas dans le tableau.' The bottom status bar shows the console output: '<terminated> RechercheBinaireValeur0Test (2) [JUnit] /Library/Java/JavaVirtualMachine'. The bottom navigation bar includes Console, Problems, and Debug Shell tabs.

7.6 TEST #6: TABLEAU NON-INITIALISÉ

Succès

Une exception « java.lang.NullPointerException » est lancée.

7.6.1 JOURNAL DE TEST

The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows a single project named "RechercheBinaireTableauNullTest".
- Run Status:** "Runs: 1/1 Errors: 0 Failures: 0" completed after 0.017 seconds.
- Code Editor:** Displays the Java code for "RechercheBinaire.java". The code includes a constructor, a private method "super()", and a public method "rechercher()". The method "rechercher()" contains a while loop that compares indices "gauche" and "droit".
- Console:** Shows the output of the test run:

```
terminated> RechercheBinaireTableauNullTest (2) [JUnit] /Library/Java/JavaVirtualMachines/dk1.8.0_181.jdk/Contents/Home/bin/java (Sep 17, 2018, 8:java.lang.NullPointerException
at ca.qc.grasset._420_pa4_ag.lab02.RechercheBinaire.rechercher(RechercheBinaire.java:27)
at ca.qc.grasset._420_pa4_ag.lab02.RechercheBinaireTableauNullTest.test(RechercheBinaire
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
```
- Breakpoints:** A sidebar showing three breakpoints labeled "Recherch" (with checkboxes checked).