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](#_Toc524946512)

[2 Analyse de la classe à tester 1](#_Toc524946513)

[3 Environnement de test 2](#_Toc524946514)

[4 Description des cas de tests 3](#_Toc524946515)

[4.1 Test #1: Élément central du tableau 3](#_Toc524946516)

[4.2 Test #2: Élément existant à droite de l’élément central 4](#_Toc524946517)

[4.3 Test #3: Élément existant à gauche de l’élément central 5](#_Toc524946518)

[4.4 Test #4: Élément inexistant à droite de l’élément central 6](#_Toc524946519)

[4.5 Test #5: Élément inexistant à gauche de l’élément central 7](#_Toc524946520)

[4.6 Test #6: Tableau non-initialisé 8](#_Toc524946521)

[5 Résultats des tests avant correctifs 9](#_Toc524946522)

[5.1 Test #1: Élément central du tableau 9](#_Toc524946523)

[5.1.1 Journal de test 9](#_Toc524946524)

[5.2 Test #2: Élément existant à droite de l’élément central 10](#_Toc524946525)

[5.2.1 Journal de test 10](#_Toc524946526)

[5.3 Test #3: Élément existant à gauche de l’élément central 12](#_Toc524946527)

[5.3.1 Journal de test 12](#_Toc524946528)

[5.4 Test #4: Élément inexistant à droite de l’élément central 15](#_Toc524946529)

[5.4.1 Journal de test 15](#_Toc524946530)

[5.5 Test #5: Élément inexistant à gauche de l’élément central 16](#_Toc524946531)

[5.5.1 Journal de test 16](#_Toc524946532)

[5.6 Test #6: Tableau non-initialisé 17](#_Toc524946533)

[5.6.1 Journal de test 17](#_Toc524946534)

[6 Description des correctifs à apporter 18](#_Toc524946535)

[7 Résultats des tests après correctifs 20](#_Toc524946536)

[7.1 Test #1: Élément central du tableau 20](#_Toc524946537)

[7.1.1 Journal de test 20](#_Toc524946538)

[7.2 Test #2: Élément existant à droite de l’élément central 21](#_Toc524946539)

[7.2.1 Journal de test 21](#_Toc524946540)

[7.3 Test #3: Élément existant à gauche de l’élément central 22](#_Toc524946541)

[7.3.1 Journal de test 22](#_Toc524946542)

[7.4 Test #4: Élément inexistant à droite de l’élément central 23](#_Toc524946543)

[7.4.1 Journal de test 23](#_Toc524946544)

[7.5 Test #5: Élément inexistant à gauche de l’élément central 24](#_Toc524946545)

[7.5.1 Journal de test 24](#_Toc524946546)

[7.6 Test #6: Tableau non-initialisé 25](#_Toc524946547)

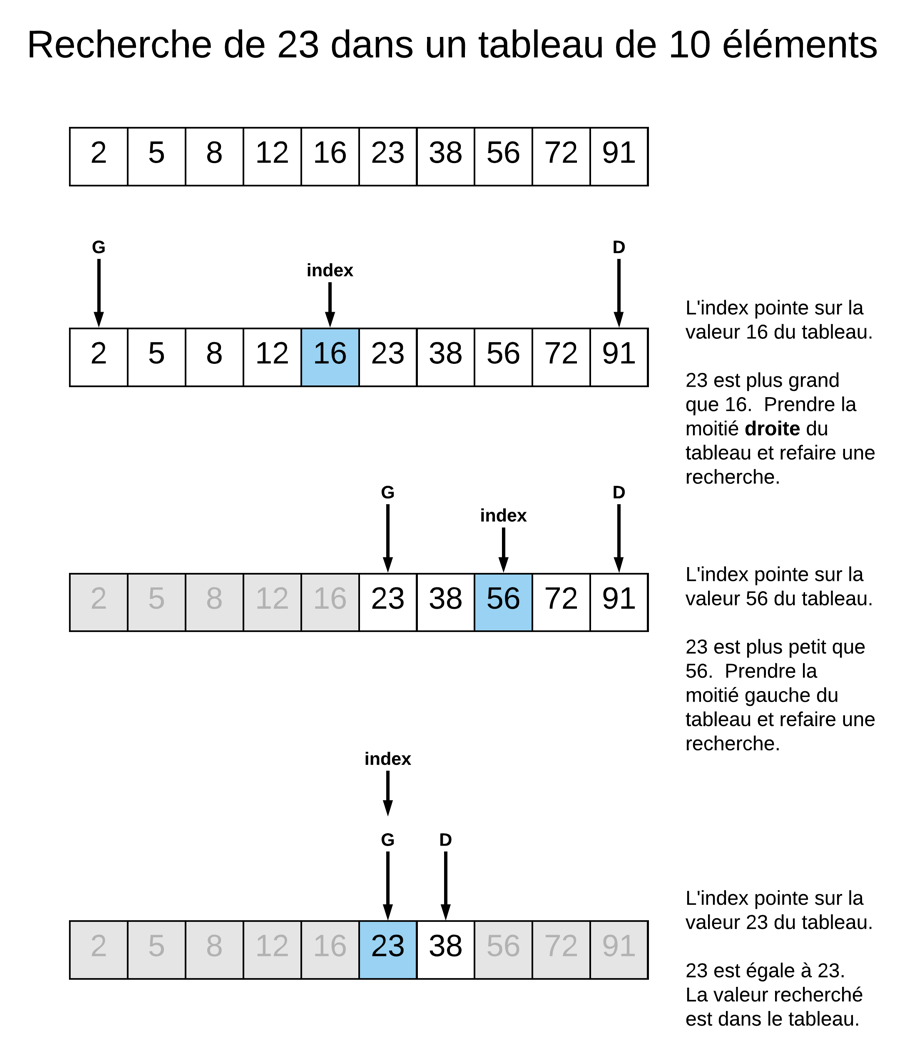
[7.6.1 Journal de test 25](#_Toc524946548)

# 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.

# 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 :



# 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 |

# Description des cas de tests

## Test #1: Élément central du tableau

|  |  |  |  |
| --- | --- | --- | --- |
| Description | Le test #1 exécute la méthode « rechercher » de la classe *Java* « RechercheBinaire » 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 :   * gauche = 0 * droit = 7   Valeur élément central : 4 |
| Post-condition | La valeur 4 est trouvée dans le tableau d’entiers. | | |

## 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 :   * 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 :   * 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 :   * gauche = 4 * droit = 4   Valeur élément central : 5 |
| Post-condition | La valeur 5 est trouvée dans le tableau d’entiers. | | |

## 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 :   * 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 :   * 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 :   * gauche = 0 * droit = 0   Valeur élément central : 1 |
| Post-condition | La valeur 1 est trouvée dans le tableau d’entiers. | | |

## 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 :   * 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 :   * 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 :   * 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 :   * 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 :   * gauche = 8 * droit = 7 |
| Post-condition | La valeur 10 n’est pas trouvée dans le tableau d’entiers. | | |

## 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 :   * 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 :   * 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 :   * 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 :   * gauche = 0 * droit = -1 |
| Post-condition | La valeur 0 n’est pas trouvée dans le tableau d’entiers. | | |

## Test #6: Tableau non-initialisé

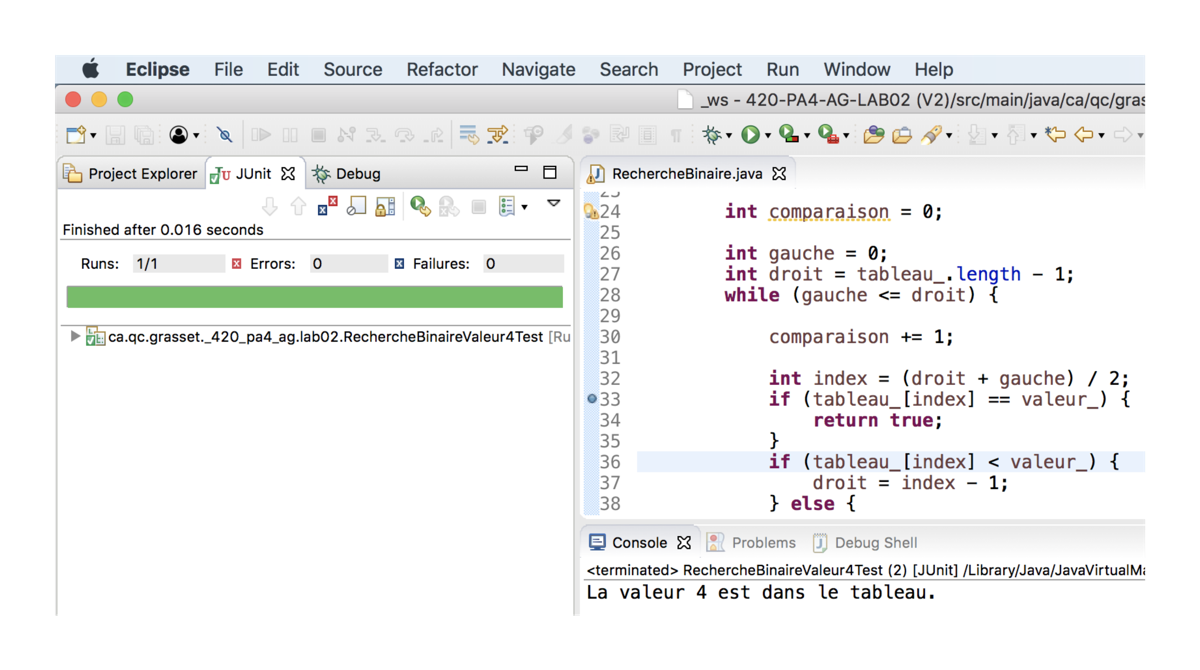
|  |  |  |  |
| --- | --- | --- | --- |
| Description | Le test #6 exécute la méthode « rechercher » de la classe *Java* « RechercheBinaire » avec un tableau d’entiers non initialisé. | | |
| Pré-conditions | Le tableau dans lequel la recherche de l’élément est effectuée n’est pas initialisé. | | |
| 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. | | |

# Résultats des tests avant correctifs

## Test #1: Élément central du tableau

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Succès** |
|  |  |  |  |
| La valeur 4 est trouvée dans le tableau d’entiers. | | | |

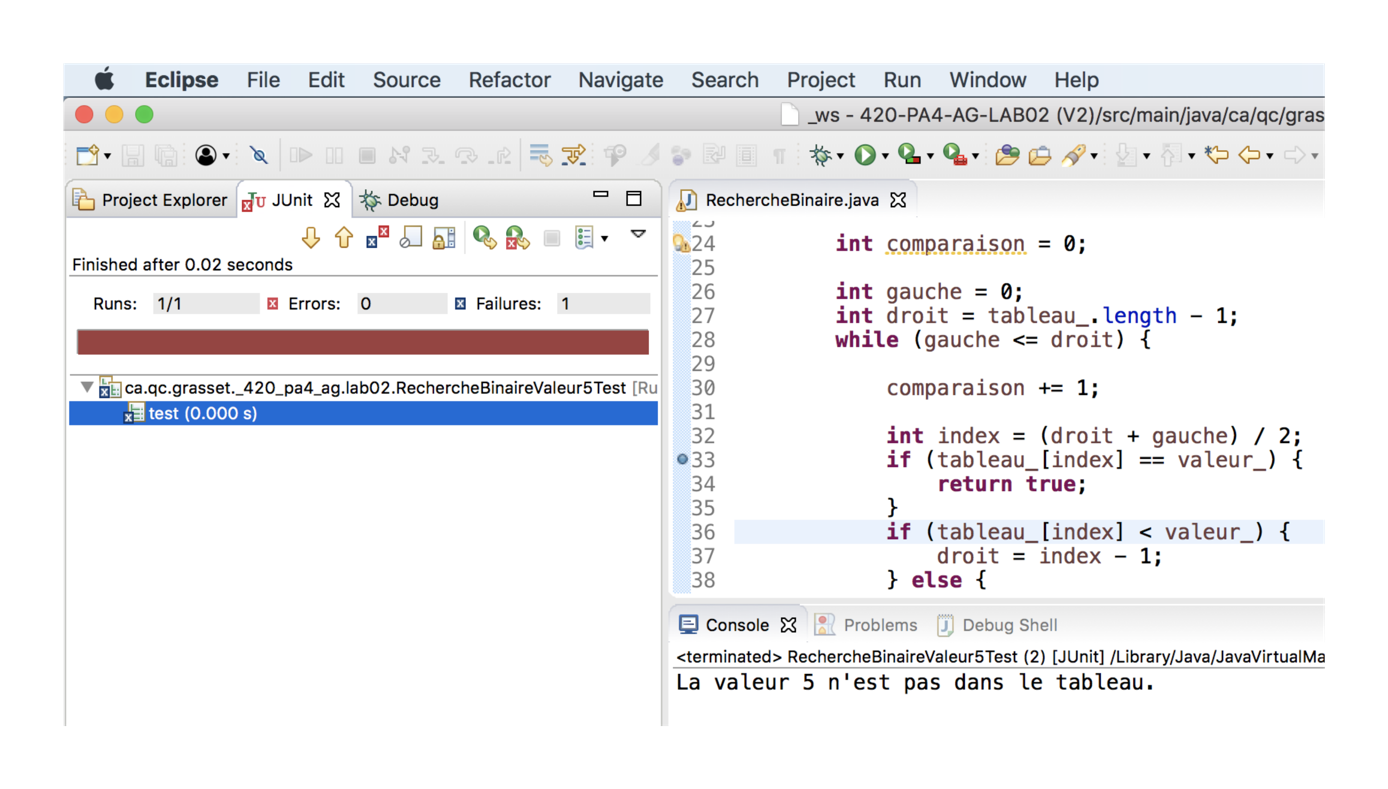
### Journal de test



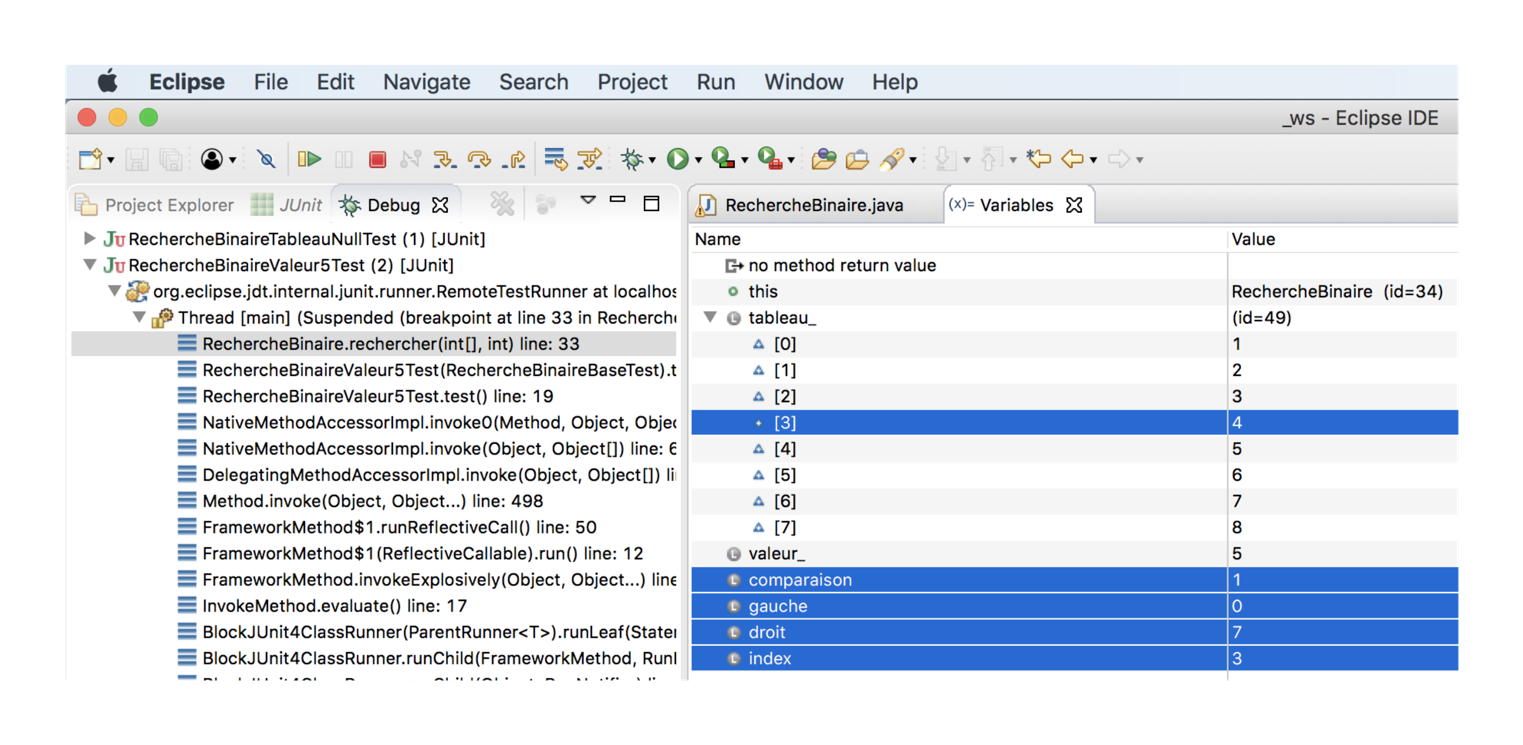
## Test #2: Élément existant à droite de l’élément central

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **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** |

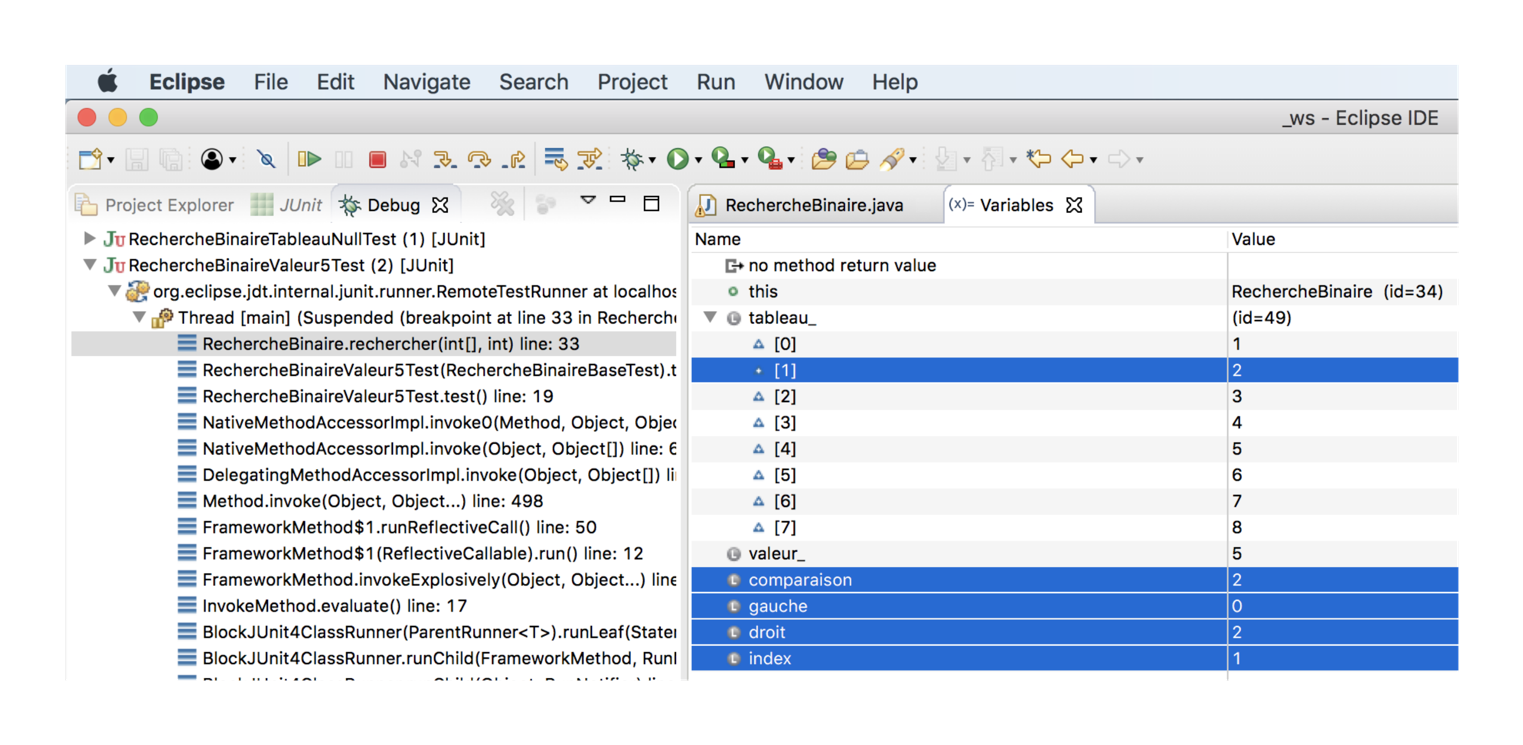
### Journal de test



#### Comparaison #1



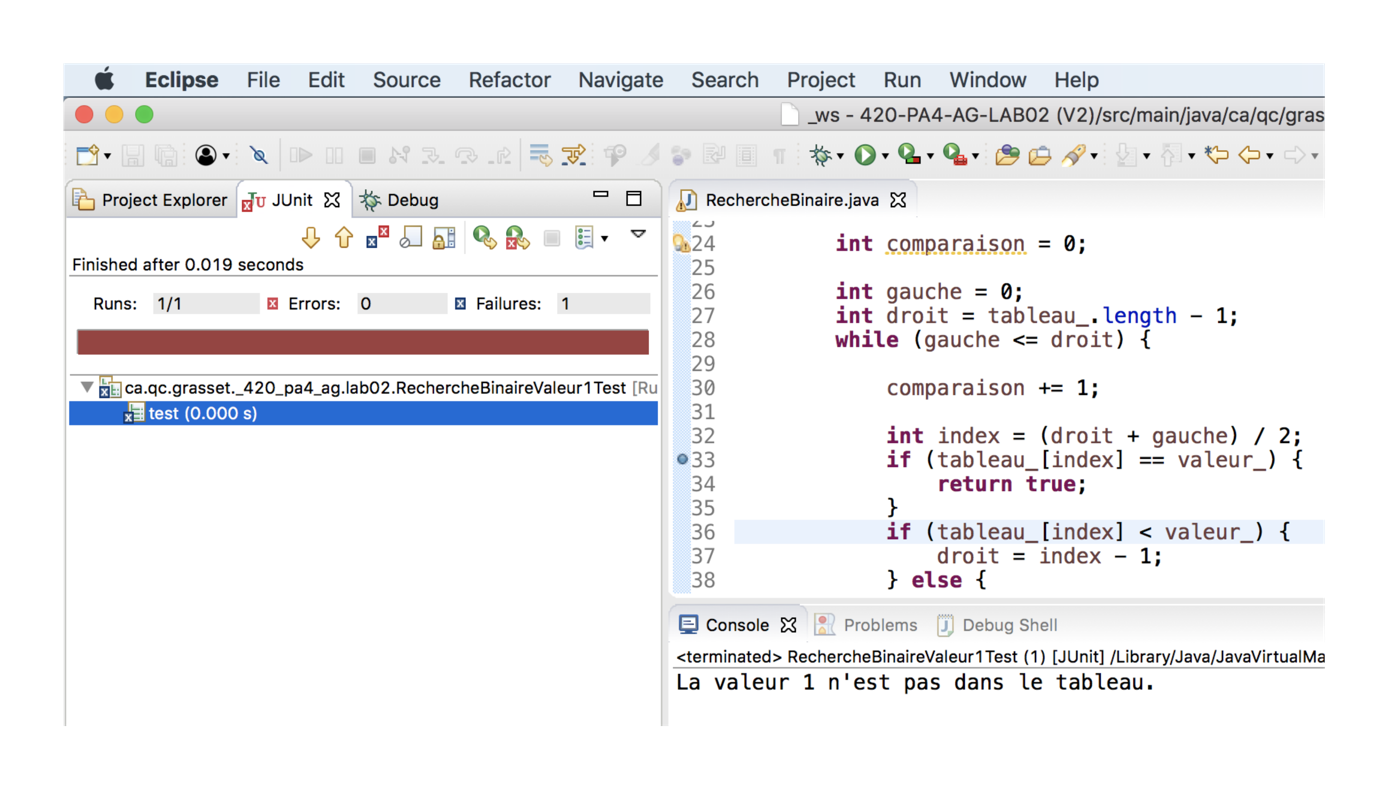
#### Comparaison #2



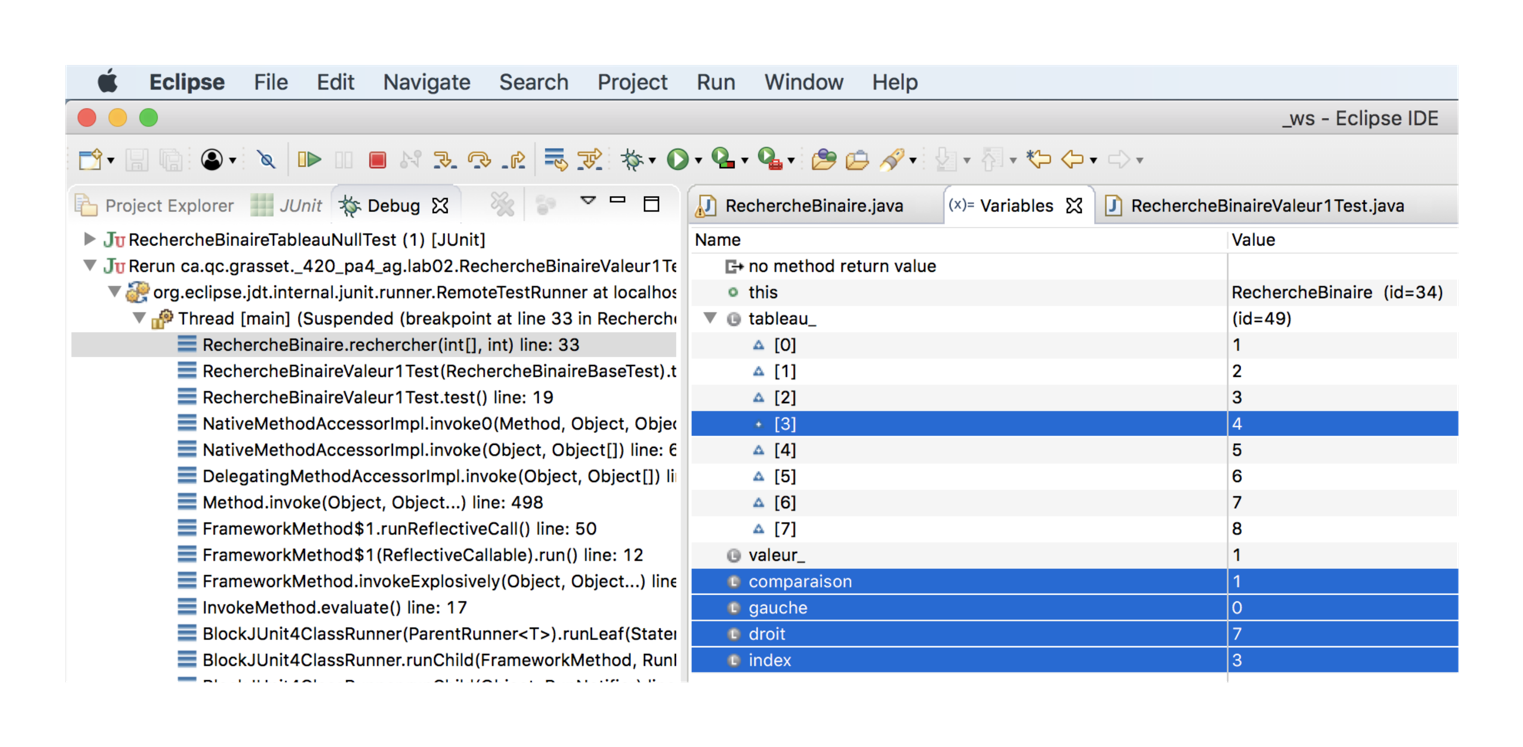
## Test #3: Élément existant à gauche de l’élément central

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **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 |

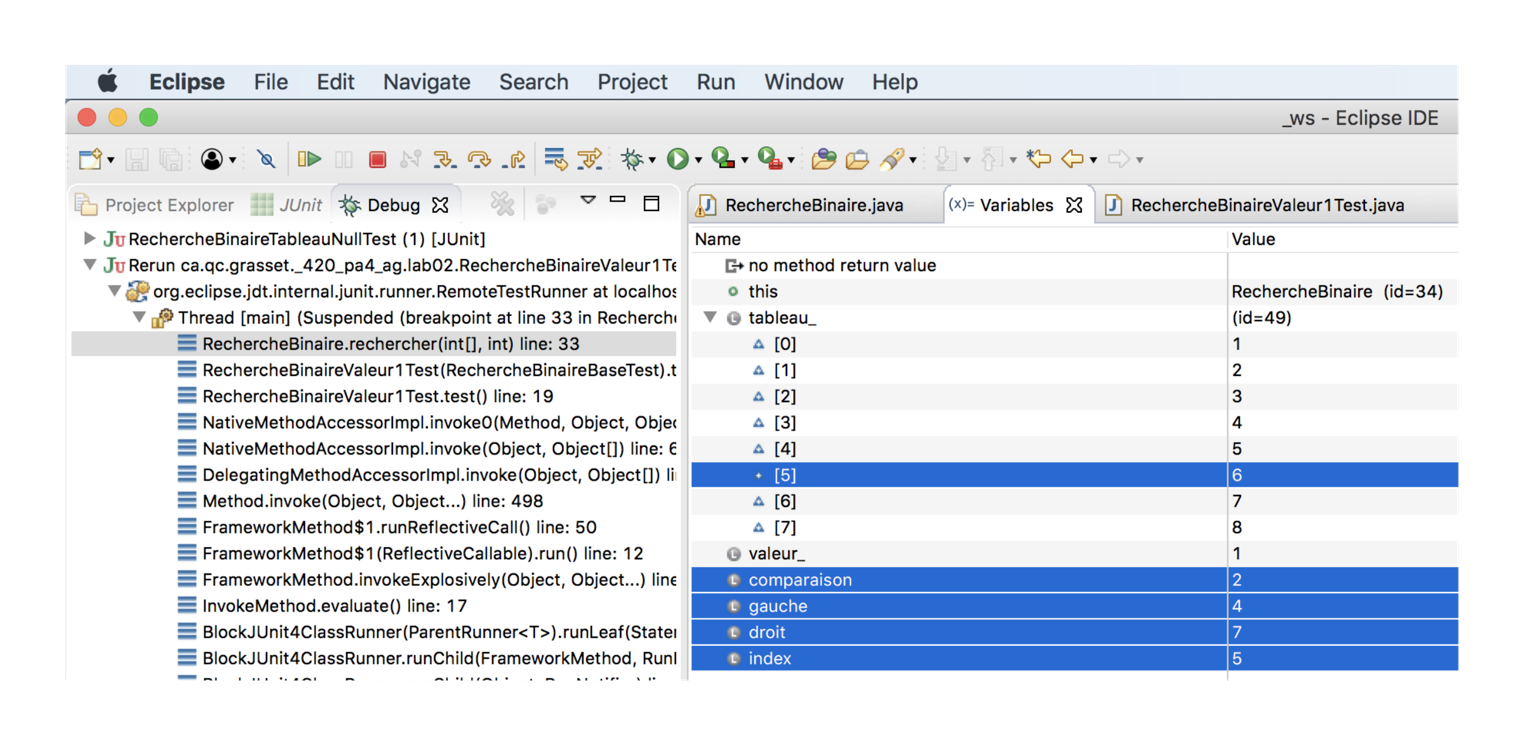
### Journal de test



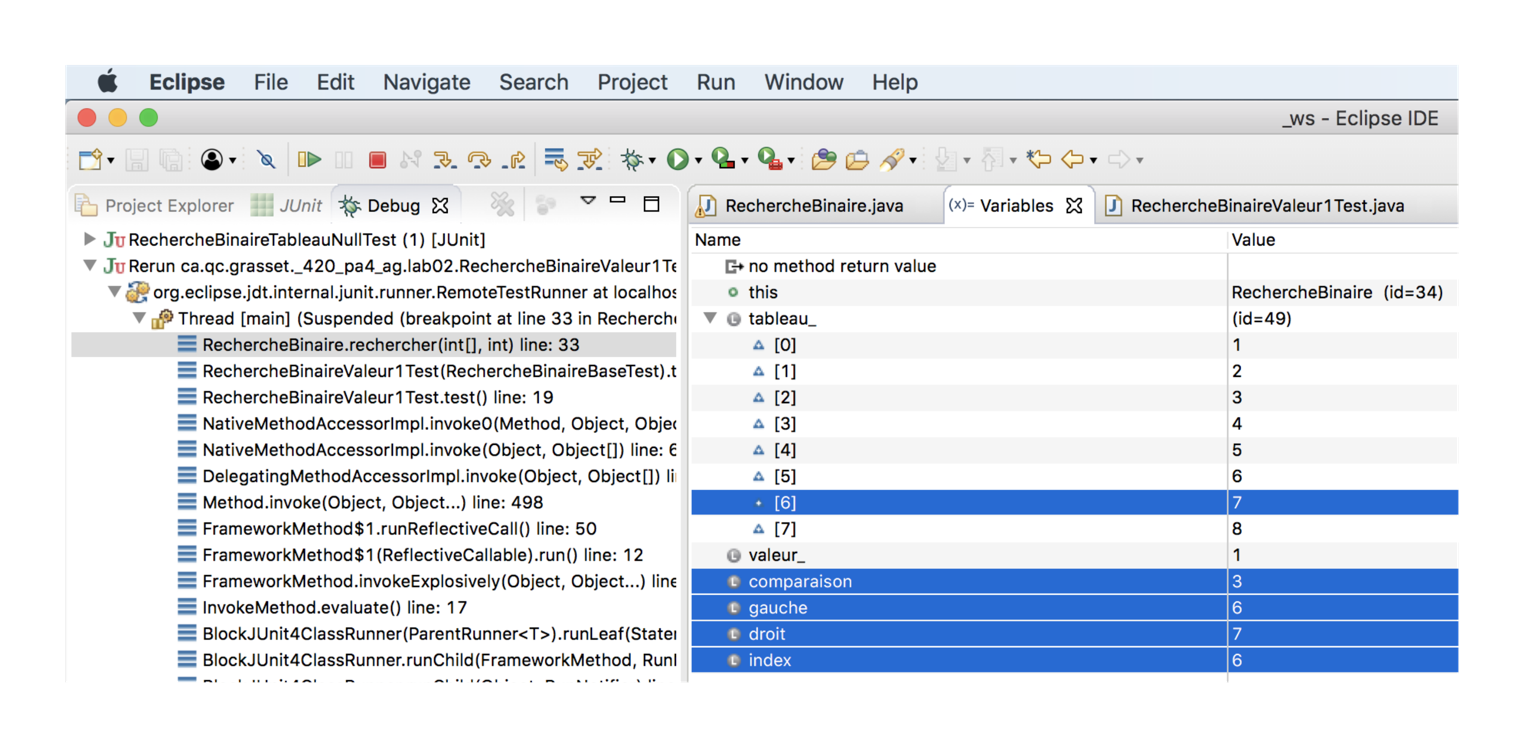
#### Comparaison #1



#### Comparaison #2



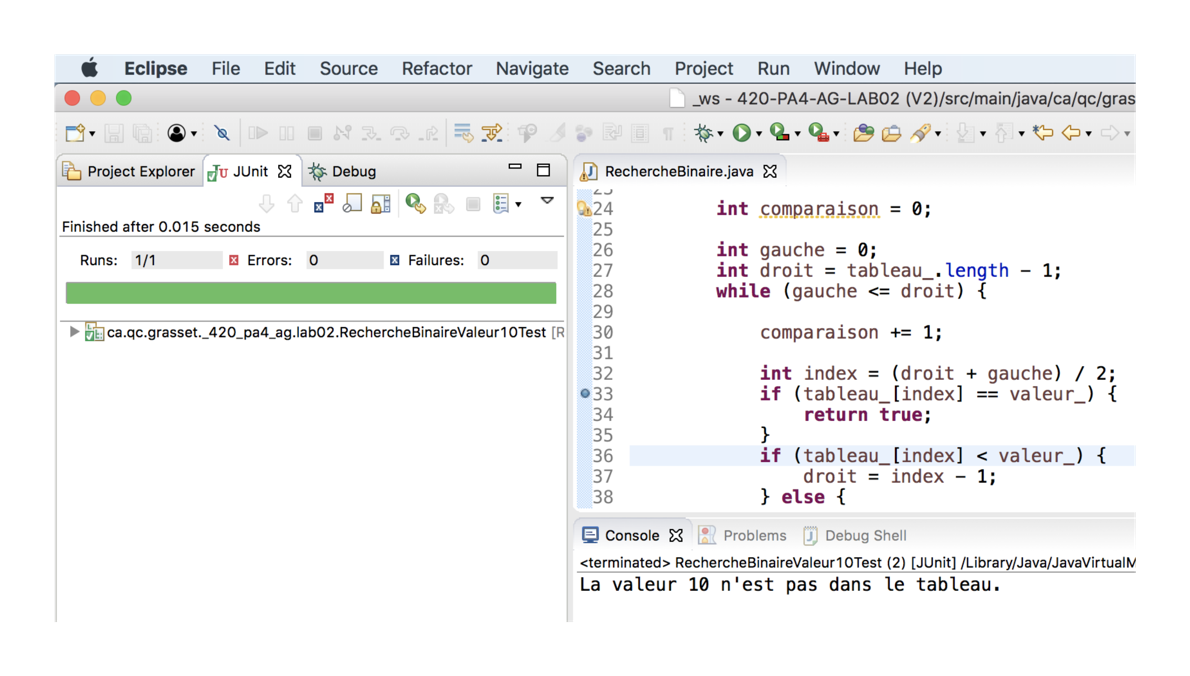
#### Comparaison #3



## Test #4: Élément inexistant à droite de l’élément central

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Succès** |
|  |  |  |  |
| La valeur 10 n’est pas trouvée dans le tableau d’entiers. | | | |

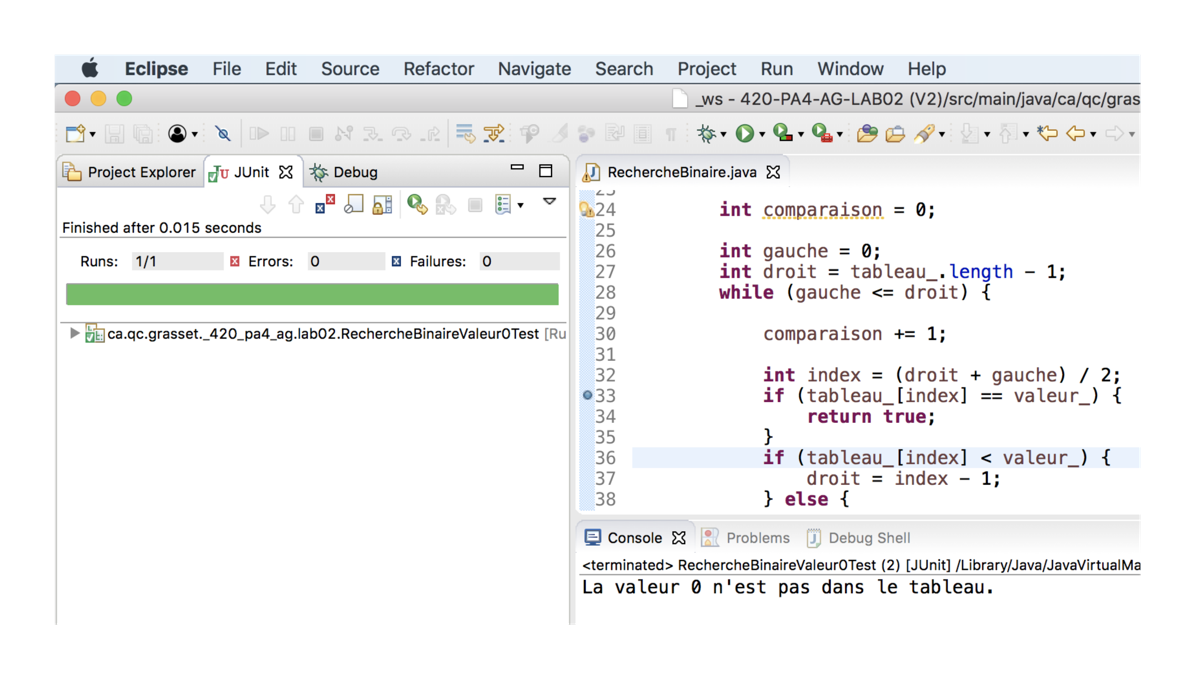
### Journal de test



## Test #5: Élément inexistant à gauche de l’élément central

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Succès** |
|  |  |  |  |
| La valeur 0 n’est pas trouvée dans le tableau d’entiers. | | | |

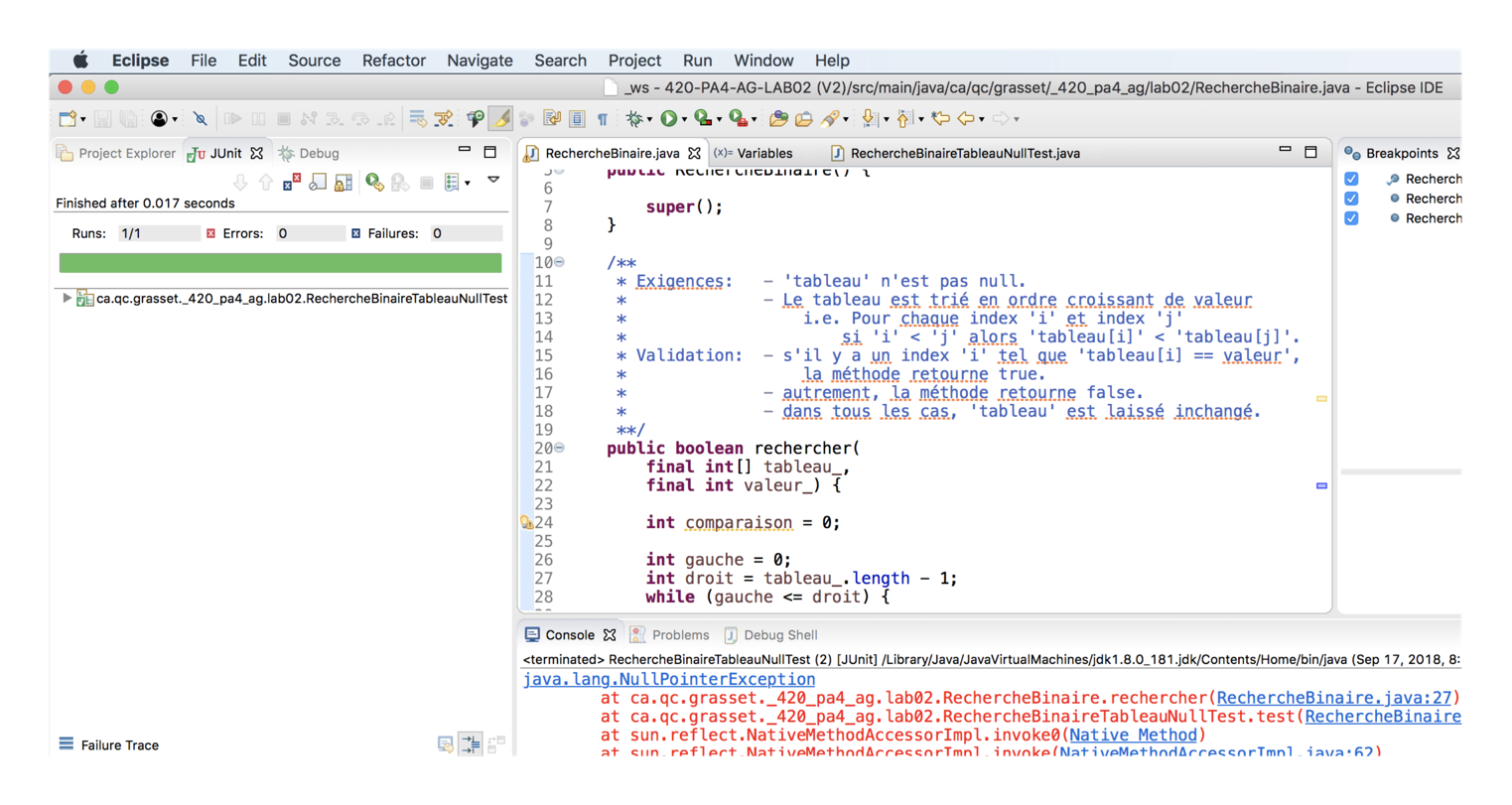
### Journal de test



## Test #6: Tableau non-initialisé

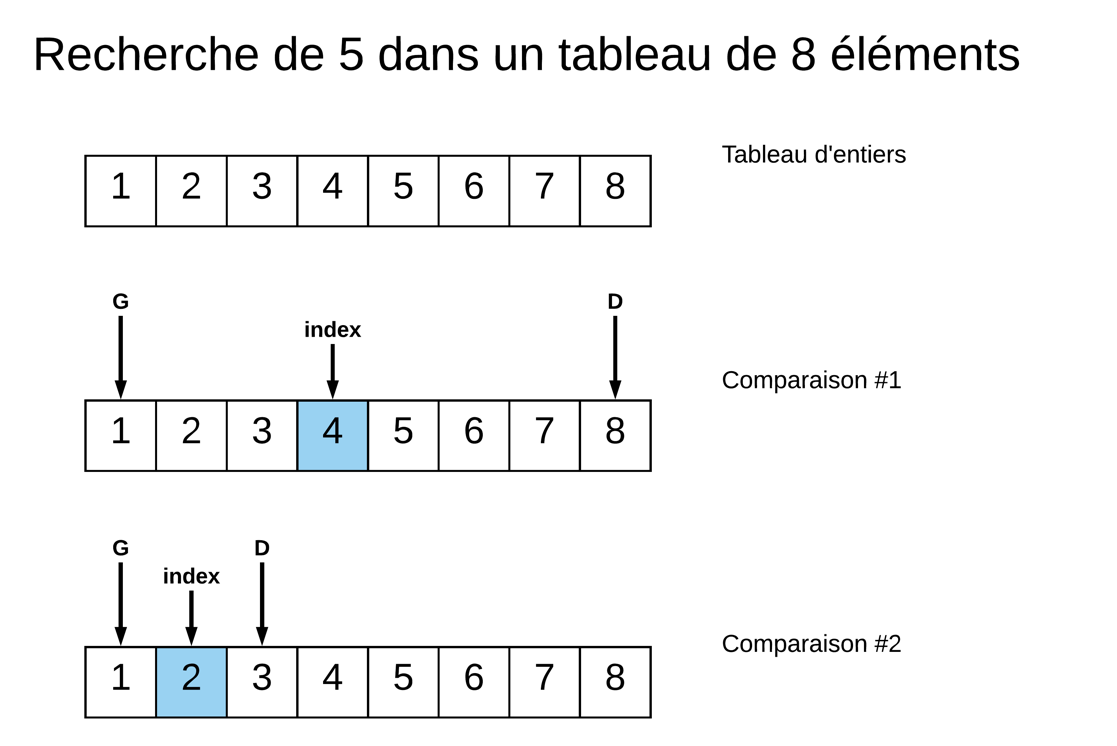
|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Succès** |
|  |  |  |  |
| Une exception « java.lang.NullPointerException » est lancée. | | | |

### Journal de test

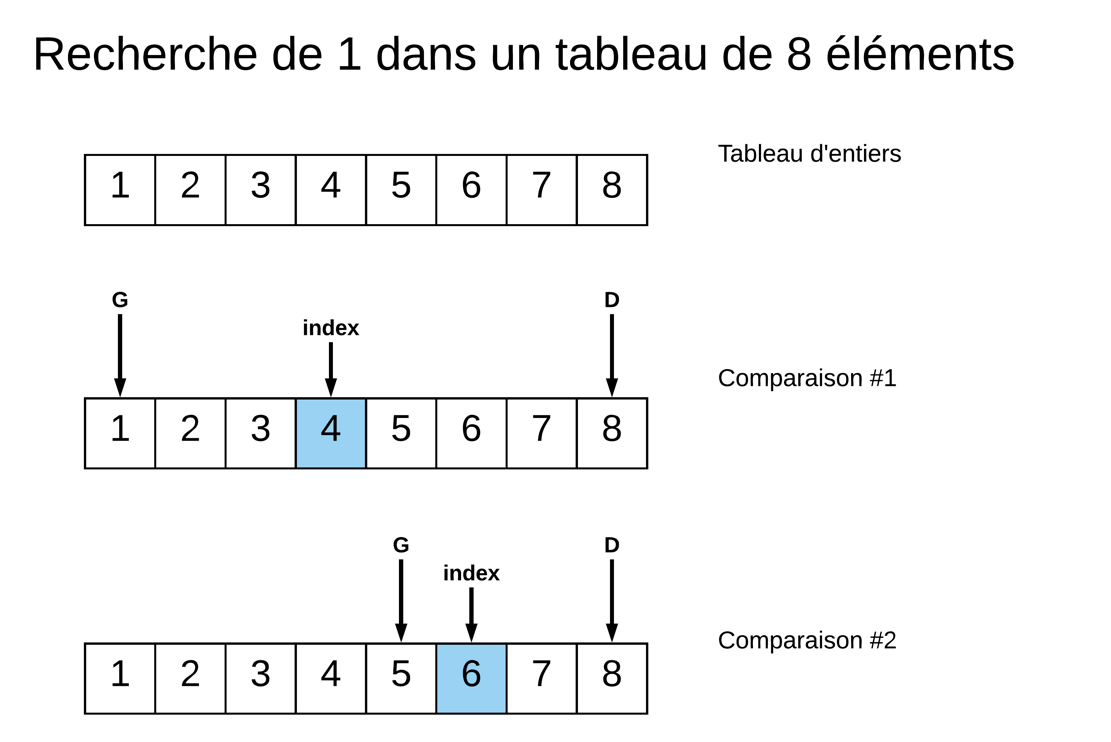


# Description des correctifs à apporter

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:



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:



L’hypothèse posée pour expliquer le problème est que l’initialisation des bornes de recherche est inversée. Le correctif proposé est donc de permuter 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:**

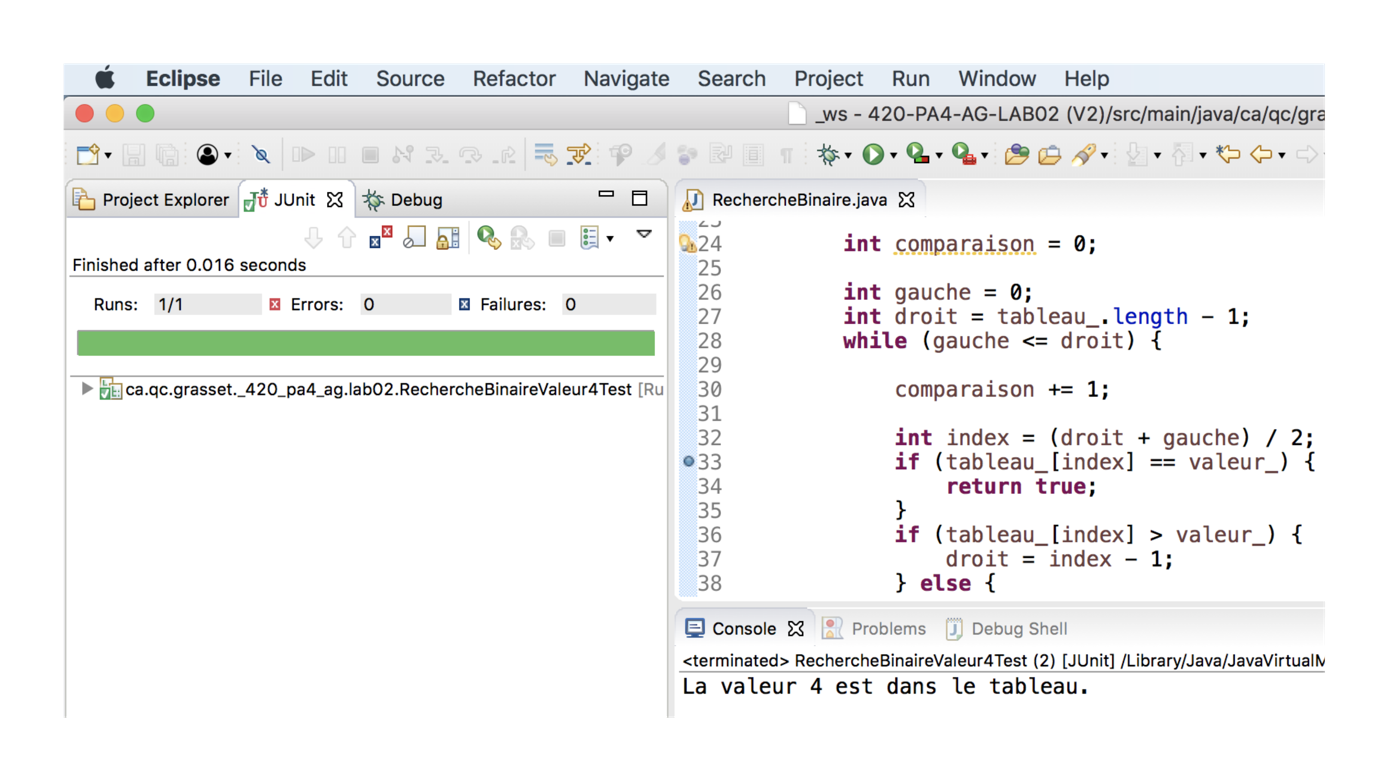
|  |
| --- |
| **public** **boolean** rechercher(  **final** **int**[] tableau,  **final** **int** valeur) {  **int** gauche = 0;  **int** droit = tableau.length - 1;  **while** (gauche <= droit) {  **int** index = (droit + gauche) / 2;  **if** (tableau[index] == valeur) {  **return** **true**;  }  **if**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

# Résultats des tests après correctifs

## Test #1: Élément central du tableau

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Succès** |
|  |  |  |  |
| La valeur 4 est trouvée dans le tableau d’entiers. | | | |

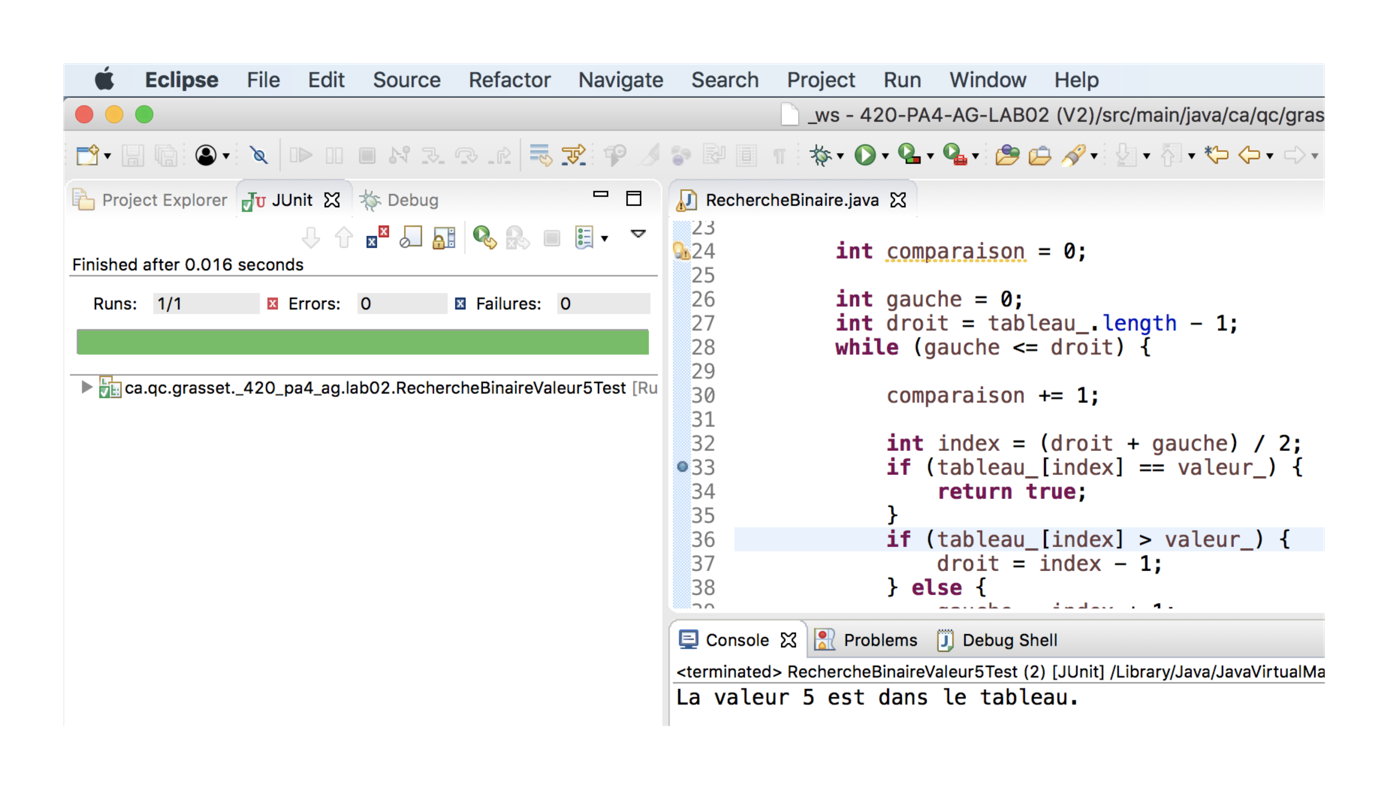
### Journal de test



## Test #2: Élément existant à droite de l’élément central

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Succès** |
|  |  |  |  |
| La valeur 5 est trouvée dans le tableau d’entiers. | | | |

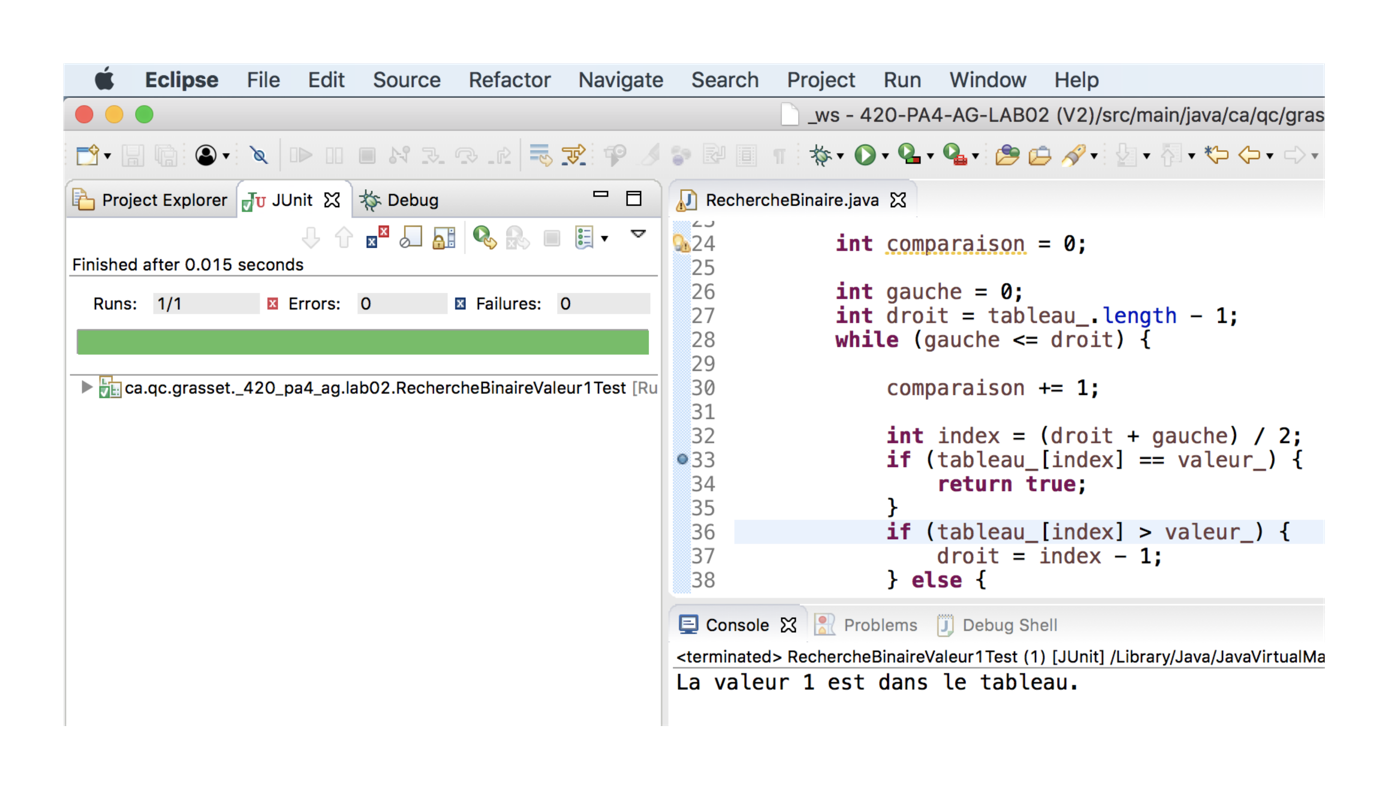
### Journal de test



## Test #3: Élément existant à gauche de l’élément central

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Succès** |
|  |  |  |  |
| La valeur 1 est trouvée dans le tableau d’entiers. | | | |

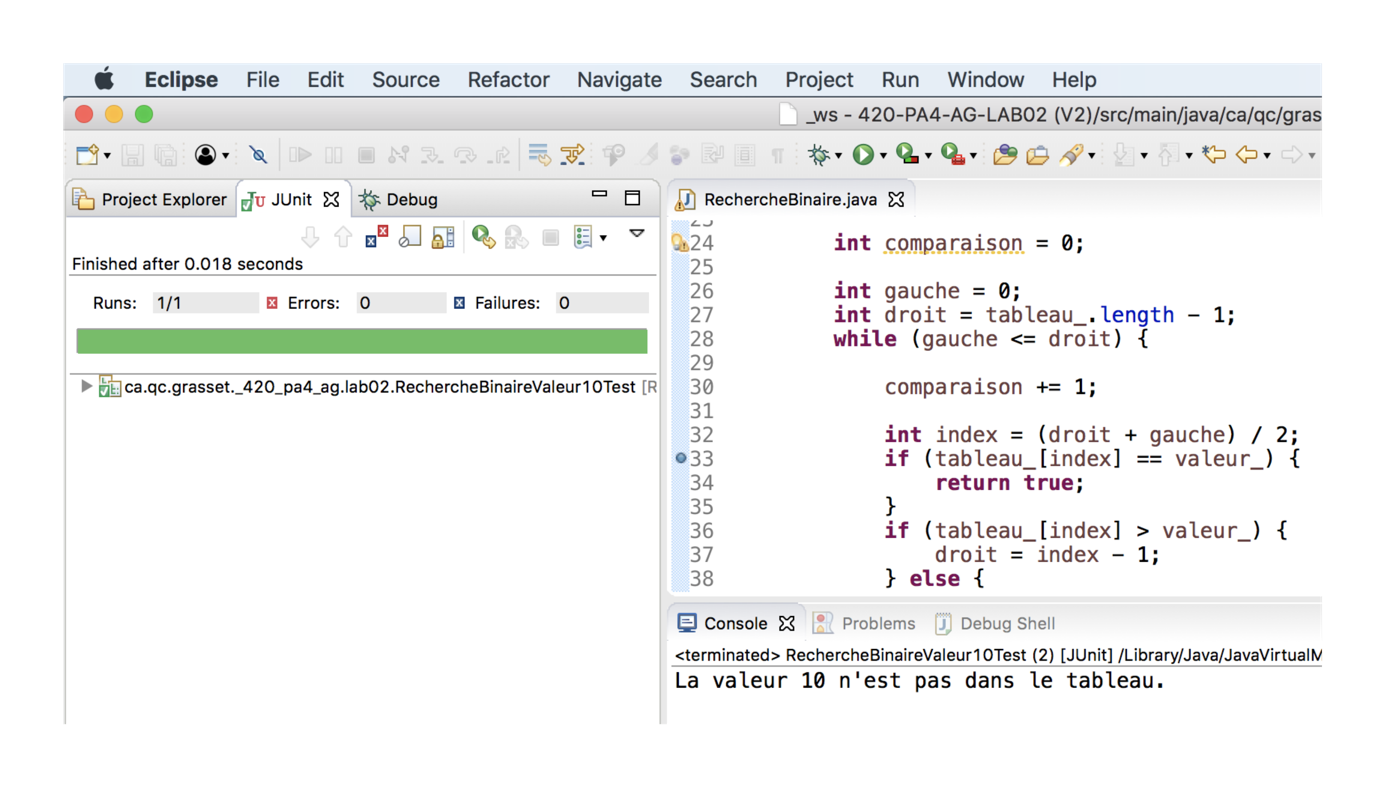
### Journal de test



## Test #4: Élément inexistant à droite de l’élément central

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Succès** |
|  |  |  |  |
| La valeur 10 n’est pas trouvée dans le tableau d’entiers. | | | |

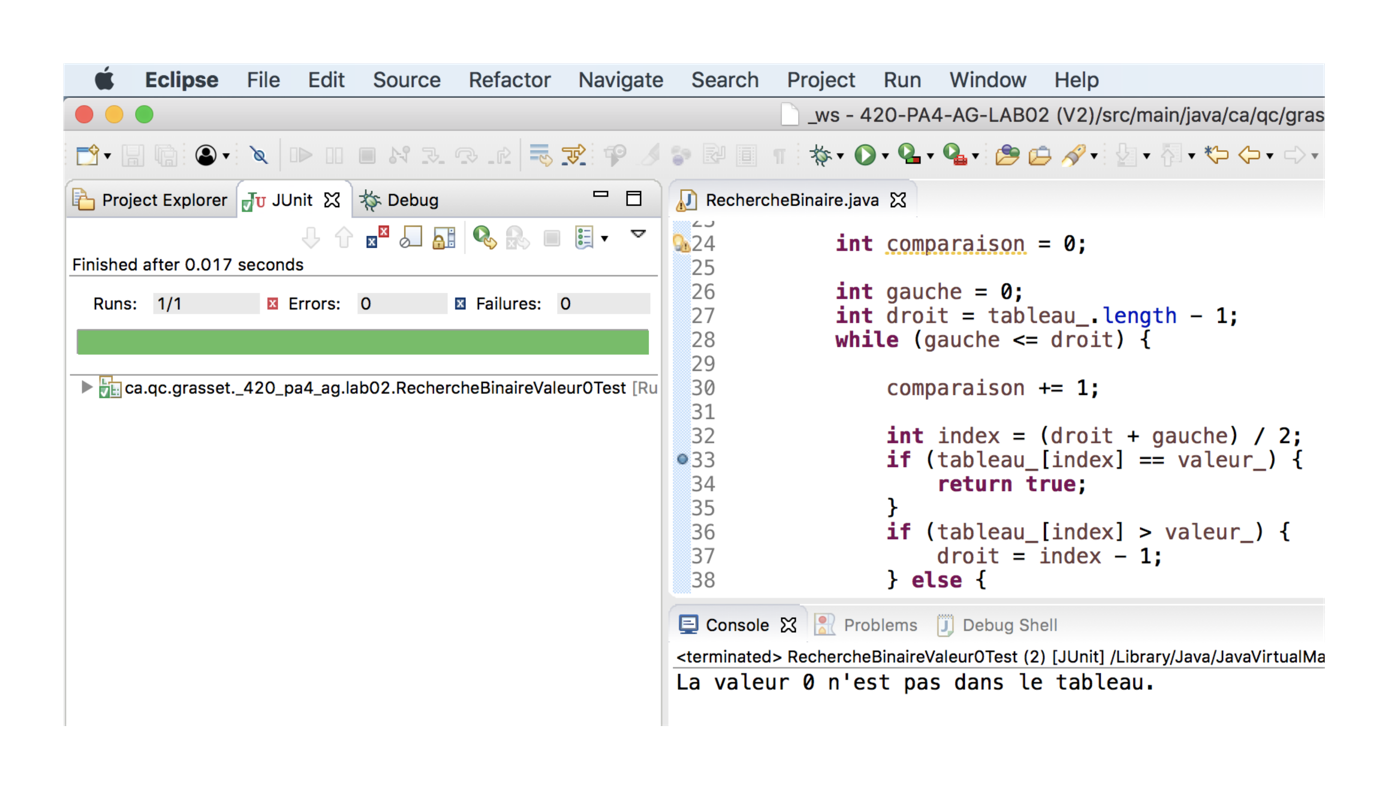
### Journal de test



## Test #5: Élément inexistant à gauche de l’élément central

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Succès** |
|  |  |  |  |
| La valeur 0 n’est pas trouvée dans le tableau d’entiers. | | | |

### Journal de test



## Test #6: Tableau non-initialisé

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  | **Succès** |
|  |  |  |  |
| Une exception « java.lang.NullPointerException » est lancée. | | | |

### Journal de test

