R4.02 - Qualité de développement
TP1

Semestre 4
2022/2023

Le code pour ce TP est disponible sur Gitlab :

`https://gitlabinfo.iutmontp.univ-montp2.fr/r402/tp1`

Depuis cette page, cliquez sur "fork" et créez votre propre dépôt avec le nom `r402-tp1-nom-prenom`, sur lequel vous travaillerez. Assurez-vous que ce dépôt est accessible à l'enseignant chargé de TP.

1 Défauts, erreurs et défaillances

Pour chacune des méthodes `indexOfLastOccurrence`, `countOddElements`, et `average` :

1. déterminez le défaut logiciel dans la méthode (si vous ne trouvez pas le défaut simplement en inspectant le code, écrivez des tests pour vous aider)
2. écrivez un test qui n'exécute pas le défaut
3. écrivez un test qui exécute le défaut, mais ne provoque pas de défaillance
4. écrivez un test qui provoque une défaillance

2 Débogueur pour l'analyse des états d'exécution

Si besoin, familiarisez-vous avec le débogueur d'IntelliJ. Un tutoriel est disponible en anglais [1] et en français [2].

En utilisant le débogueur, et sans modifier le code source de la méthode `f`, répondez aux questions suivantes :

1. Si on exécute la méthode `f` avec la valeur `n` = 18, quelles sont les valeurs successives prise par la variable `i` ?

1. `https://blog.jetbrains.com/idea/2020/05/débogueur-basics-in-intellij-idea/`
2. `https://jetbrains.developpez.com/tutoriel/jetbrains-introduction-debogueur-intellijidea/`

2. Si on exécute la méthode `f` avec la valeur `n` = 31, le programme atteindra un état où, au début de la boucle, la variable `i` vaut 7. Dans cet état, quelle est la valeur de la variable `j` ?
3. Si la méthode `f` passe par un état où `i` = 20 et `j` = 3 (en début de boucle), quelle valeur sera retournée à la fin de l'exécution ?

3 Débogueur pour l'analyser d'une application complexe

Exécutez la méthode `ecommerceUsecase` et répondez aux questions suivantes en utilisant le débogueur :

1. Donnez les valeurs des différents champs de l'objet (`Client`) `c` immédiatement après sa création. Si certains de ces champs sont des objets, listez aussi les valeurs de leurs champs.
2. Donnez sous forme d'arbre (arbre d'appel) toutes les méthodes qui sont exécutées par l'appel à `add(p)`.

4 Débogueur pour la réparation de défauts logiciels

1. Écrivez des tests pour la méthode `sort`, en essayant de choisir des entrées adaptées pour découvrir des défauts logiciels.
2. Utilisez le débogueur pour trouver les défauts logiciels dans les méthode `sort`, `sortSegment` et `mergeSortedArrays`, puis réparez-les. Pour chaque défaut trouvé, décrivez le problème, indiquez la modification effectuée pour le résoudre, et listez les tests dont le statut a changé suite à la modification.
3. Pouvez-vous déterminer quand tous les défauts ont été trouvés ? Expliquez pourquoi, et imaginez des moyens pour mesurer la confiance qu'on peut accorder aux tests.