Activté 1 C306 - Ingénierie du logiciel

Session 2021 - Semestre 2

Consortium International e-Miage - Auteur: *Sébastien Choplin* sebastien.choplin@u-picardie.fr

Le devoir déposé sur la plate-forme doit être une archive (zip) contenant: un fichier au format PDF synthétisant le travail (copier-coller des codes et remarques éventuelles) les différents codes développés et rapports générés.

Vous devrez également déposer ces documents sur un dépôt GIT ou SVN en fonction du choix de votre tuteur. Votre dépôt resservira pour les autres activités.

Exercice 1 : Ecriture de code

```java
/** Classe Produit représentant un produit avec un prix et une référence */
public class Produit {
 private final String reference;
 private final double prix;
 double tva = 0.20;

public Produit(String reference) {reference = reference;}

/** @return le prix */
public double getPrix() {
return this.prix;
}

/** modifie le prix */
public final void setPrix(Double prix){this.prix = prix;}

/**
* @return la reference si le prix est positif, null sinon
*/
public final String getReference() {
    String resultat = reference;
if ( prix > 0) return reference;
}
        @Override
    public boolean equals(Object o) {
    return reference == ((Produit)o).reference;
    }
}
```

1. Pourquoi ce code ne compile-t-il pas ?
2. Corrigez le code pour qu'il compile (en respectant les consignes de la documentation).
3. Donnez le rapport de *checkstyle*, *spotbugs* et *PMD* appliqués à votre code.
4. Proposez une réécriture en tenant compte des problèmes soulevés par *checkstyle*, *spotbugs* et *PMD*.
5. Donnez les rapports appliqués au code réécrit.

Ressources CheckStyle

- https://checkstyle.sourceforge.io/

Ressources SpotBugs

- https://spotbugs.github.io/
- https://spotbugs.readthedocs.io/en/stable/running.html

  usage: (produit un fichier de rapport au format *html*)

  ```
  spotbugs-4.3.0$ java -jar lib/spotbugs.jar -html -output rapport-spotbugs.html Produit.class
  ```

Ressources PMD

- https://pmd.github.io/

Exercice 2 : Tests unitaires

Dans cet exercice, vous allez implémenter des algorithmes sur un tableau d'entiers. Le but est de fournir un code respectant les règles d'écriture et accompagné d'un jeu de tests unitaires permettant de valider qu'il fourni le résultat attendu.

Copier du code trouvé tout fait n'a aucun intérêt : le but de l'exercice est d'écrire un programme, les tests qui vont avec et valider la bonne écriture du programme. **Les méthodes de la classe "Arrays" sont interdites.**

```java
public final class TabAlgos {

  /** @return valeur la plus grande d'un tableau. */
  public static int plusGrand(final int[] tab) {
   // TODO
  }

  /**
   * @return moyenne des valeurs du tableau.
   * @throw IllegalArgumentException si tab et null ou vide.
   **/
  public static double moyenne(final int[] tab) {
   // TODO
  }

  /**
   * Compare le contenu de 2 tableaux en tenant compte de l'ordre.
   * @return true si les 2 tableaux contiennent les mêmes éléments
   *         avec les mêmes nombres d'occurences
   *         (avec les elements dans le meme ordre).
   **/
  public static boolean egaux(final int[] tab1,final int[] tab2) {
   // TODO
  }

  /**
   * Compare le contenu de 2 tableaux sans tenir compte de l'ordre.
   * @return true si les 2 tableaux contiennent les mêmes éléments
   *         avec les mêmes nombres d'occurrence
   *         (pas forcément dans le meme ordre).
   **/
  public static boolean similaires(final int[] tab1,final int[] tab2) {
   // TODO
  }
}
```

Pour chacune des questions, fournir le code et les rapports correspondants, dans une arborescence adéquate pour permettre la re-exécution des différentes étapes.

1. **Ecrivez des tests unitaires** (en utilisant JUnit 5) permettant de tester les méthodes à implémenter.
2. **Implémentez les méthodes** en respectant les règles d'écriture contrôlées par les outils *checkstyle*, *spotbugs* et *PMD*.
3. **Vérifiez la validité des tests** avec le code implémenté.
4. **Fournir les rapports** des tests unitaires, de *checkstyle*, *spotbugs* et *PMD*. **Il ne doit rester aucune anomalie.**

Exercice 3 : Dépôt sur serveur de Versionning

Suivant le choix de votre tuteur, vous allez utiliser un serveur GIT ou un serveur SVN.

Solution GIT

- Installez git
- Paramétrez git et indiquez le nom et l'email que vous avez utilisé.
- Générez votre paire de clefs RSA : clef publique / privée (ou ré-utilisez la votre si vous en avez déjà une).
- Communiquez votre clef au publique à votre tuteur qui l'utilisera pour vous donner accès à votre dépôt distant GIT. Votre tuteur vous donnera l'adresse du dépôt.
- Déposez le travail de cet activité dans votre dépôt, **dans le dossier activite1**.

Solution SVN

- Installez un client SVN.
- Demandez vos paramètres d'accès et l'adresse du dépôt à votre tuteur.
- Récupérez une copie de votre dépôt distant.
- Déposez le travail de cet activité dans votre dépôt, **dans le dossier activite1**