Révision TQSI

Juin 2020

I. Questions

A) Tests et analyses

1. Expliquez brièvement les différences entre tests unitaires, tests d'intégration et tests fonctionnels.
2. Expliquez ce qu'est un mutant et un score de mutation des tests.
3. Donnez deux critères de couverture du code par vos tests
4. Vos tests sont meilleurs si (cochez la ou les bonnes réponses):
   - ✔ A : ils sont en majorité passants
   - ✔ B : ils sont en majorité non passants
   - ✔ C : ils ont un grand taux de couverture
   - ✔ D : ils ont un faible taux de couverture
   - ✔ E : ils fournissent un grand score de mutation
   - ✔ F : ils fournissent un petit score de mutation
5. On souhaite évaluer la qualité du projet. Quelle est la différence entre les techniques d'analyse statique de code et les techniques d'analyse dynamique de code.
6. Classifiez les outils suivants selon de type d'analyse effectuée (statique ou dynamique) : PMD, COBERTURA, CHECKSTYLE, JUNIT, PIMUTATION, EMMA

B) Systèmes de gestion de versions

1. Deux utilisateurs ali et saida modifient leur copie locale d'un fichier du dépôt SVN. Après modification, chacun poste le fichier dans le dépôt. Que se passe-t-il dans ce cas ?
2. Expliquez la gestion du problème de conflit entre versions locale et distante d'un fichier dans un outil SCM comme SVN

C) Plate-forme d'intégration continue et qualité

1. Donnez le cycle de fonctionnement d'une plate-forme comme JENKINS
2. Est-il possible configurer JENKINS afin de commencer son cycle après chaque commit ?

D) Tests de montée en charge

1. Quelle est l'intérêt des tests de montée en charge ?
2. Avant de lancer un test de montée en charge, on doit d'abord obtenir son scénario. Expliquez comment faire pour enregistrer le scénario d'un test de montée en charge avec un outil de test comme JMETER et faites un schéma.

II. Ex 1

Soit une application «DistributeurCafe», la spécification de cette application permet les opérations suivantes :

- Ajout inventaire : un inventaire consiste en quantités d'ingrédients utilisés dans la préparation des boissons. L'inventaire est ajouté par cette opération à la quantité existante. Un inventaire diminue uniquement lors de l'achat d'une boisson.
- Achat boisson : si la boisson existe et si l'acheteur a suffisamment d'argent, il peut prendre une boisson et récupérer sa monnaie.

Note : vous donnez les tests uniquement, n'écrivez pas les classes du distributeur café qu'on suppose existantes.

Voici un extrait de la classe **DistributeurCafe** :

```
public class DistributeurCafe {
   public  void ajouterInventaire(int qteCafe, int qteLait, int qteSucre, int
                                                    qteChocolat) throws InventaireException

   /** prearation cafe au client. Le type du cafe : 0 pour cafe au lait, 1 pour cafe
   noir, 2 pour cafe cappucino, 3 pour cafe chocolat */
   public  int preparerCafe(int type, int qtePayee)
}
```

Créez des tests d'acceptations selon les exigences suivantes :

1. Créez une classe de tests InventaireTest1 contenant le test suivant
   - ✔ **ajouterInventaireExceptionTest** : ici on vérifie la levée d'une exception lors de l'ajout d'inventaire
2. Créez une classe de tests InventaireTest2 contenant le test suivant
   - ✔ **ajouterInventaireTest** : ici on vérifie qu'aucune exception n'est levée
3. Créez une classe de tests CafeTest contenant le test suivant
   - ✔ **preparerCafeTest** : ici on vérifie que le café est préparée en moins de 30 s et qu'on retourne bien la monnaie correcte de l'achat
4. Comment connaître la raison de l'échec de test ?

5. Quelle est la manière la plus simple pour lancer l'ensemble des tests précédents ?

III. Ex 2

Donnez le contenu d'un fichier build ANT permettant de :

- ✔ compiler et exécuter la classe DistributeurCafe
- ✔ exécuter la classe DistributeurCafe
- ✔ compiler et exécuter tous les tests
- ✔ générer les rapports html des tests

Vous utiliserez l'arborescence suivante :

- ✔ **src** : sources de l'application
- ✔ **test** : sources des tests
- ✔ **classes** : les classes compilés
- ✔ **data** : les fichiers intermédiaires en formats texts, xml, …
- ✔ **report** : les fichiers html