IN506 - Spéc. formelles - Examen

Stéphane Lopes

2h - Documents autorisés

Exercice 1
Répondez de façon **précise et concise** aux questions suivantes.

1. A quoi sert l'outil *CheckStyle* ? Comment l'intègre-t'on dans un processus de développement ?
2. Quelles différences y-a-t'il entre *Ant* et *Maven* ?
3. Donnez la cible *Ant* pour exécuter *Javadoc* sur le projet. Même question pour exécuter *Checkstyle*.
4. Quelle commande *Maven* permet de générer une archive du projet ?
5. Sous *Maven*, quelles relations y-a-t'il entre *cycle de vie*, *phase*, *plugin* et *but* ?
6. Que doit-on faire pour ajouter une dépendance vers JUnit 4.10 le POM ? Les coordonnées du projet JUnit 4.10 sont : `groupId = junit`, `artifactId = junit`, `version = 4.10`.
7. Dans un VCS, qu'est-ce qu'une *branche* ? Qu'est-ce qu'un *tag* ?
8. Donner la commande *Subversion* permettant d'obtenir une copie de travail du tronc du projet se trouvant à l'URL `file:///svnrepos/projet1`.
9. Dans un VCS, quelle commande permet de valider un ensemble de modification ? Donnez un exemple avec *git*.
10. Pourquoi un ensemble de tests ne permet-il pas de prouver l'absence de bogue dans un programme ?
11. Qu'est-ce que le *refactoring* ? Donnez un exemple.

Exercice 2
Soit le code Java implémentant l'addition de nombres complexes ci-dessous :

```
public Complex add(Complex addend) throws NullArgumentException {
  MathUtils.checkNotNull(addend);
  if (isNaN || addend.isNaN) {
    return NaN;
  }
  return createComplex(real + addend.getReal(),
                       imaginary + addend.getImaginary());
}
```

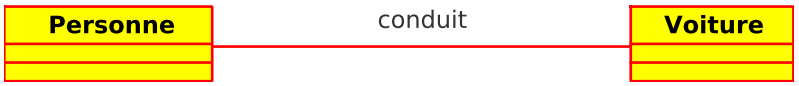
Vous supposerez que :
- la classe `Complex` possède un constructeur `Complex(double real, double imaginary)`,
- il existe une constante `Complex.NaN` (*Not A Number*).

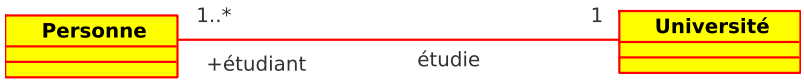
Ecrivez les tests JUnit 4 suivants :

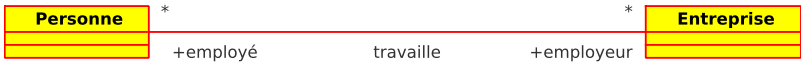
1. $(1 + 2i) + (3 + 4i) = (4 + 6i)$,
2. donnez le *test fixture* créant les objets utilisés dans les différents tests,
3. $(1 + 2i) +$ `Complex.NaN` $=$ `Complex.NaN`,
4. `Complex.NaN` $+ (3 + 4i) =$ `Complex.NaN`,
5. $(1 + 2i) +$ null lève une exception de type `NullArgumentException`.

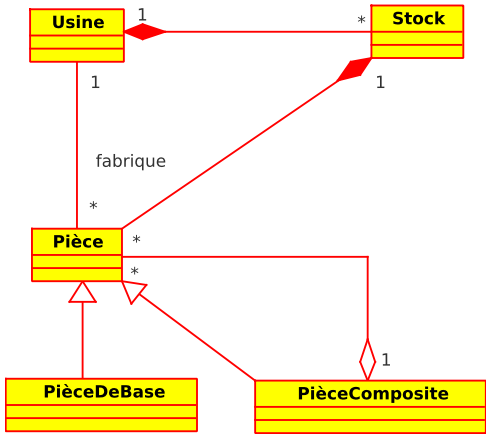
Tourner la page SVP →

Exercice 3
Donner la signification en français des modèles UML suivants :

1. 

2. 

3. 

4. 

Exercice 4
On veut implémenter une bibliothèque pour la manipulation d'expressions arithmétiques.

Une expression arithmétique est un arbre dont les nœuds sont des constantes numériques, des opérateurs binaires ou des fonctions unaires (prédéfinies). Une constante représente un nombre en virgule flottante. Un opérateur binaire possède un symbole et deux opérandes qui sont elles-mêmes des expressions. Une fonction possède un nom et s'applique sur une expression.

La bibliothèque doit fournir les services suivants :
- création d'expression (par programme, i.e. on ne développe pas d'analyseur syntaxique),
- évaluation d'expression,
- affichage d'un expression.

1. Proposez un diagramme de classes UML détaillé (méthodes, multiplicité, ...) représentant cet énoncé.
2. Pour l'expression $2.0^2 - 4.0 \times 3.0 \times 5.0$,
   (a) donnez le diagramme d'objet de l'expression,
   (b) donnez les diagrammes de séquence et de communication pour l'évaluation de l'expression.