Université Paris Diderot **Programmation Orientée Objet**
L2/L3 **Année 2014-2015**

TD - Séance n°2

Révisions – Classes, Modélisation

Lisez attentivement le sujet, les exercices sont toujours plus faciles à faire quand on a *bien* lu l'énoncé. Parfois il peut être utile de lire les questions suivantes : comme ça on sait où l'exercice veut en venir !

Les exercices de la premiere partie du TD doivent tous être traités avant la semaine prochaine. Finissez ceux que vous n'avez pas eu le temps de faire en TD chez vous. Les exercices de la seconde partie sont à faire si vous vous sentez à l'aise.

1 Exercices obligatoires

Exercice 1 *Questions de cours*

1. Quand est-il intéressant d'avoir une fonction `static` ?
2. Définissez la surcharge de fonction en Java. En quoi est-ce utile ?
3. Qu'est-ce que `null` en Java ?
4. Expliquez la commande `System.out.println("texte")`

Exercice 2 *Évaluation de code*

Qu'affiche le code suivant ? Pourquoi ?

```
class A {
    private int attr;

    A(int value_attr) {
        this.attr = value_attr;
    }

    public bool egal(A b) {
        if(this.attr == b.attr) {
            return true;
        }
        else {
            return false;
        }
    }
```

```
        public int getAttr() {
            return this.attr;
        }
    }
```

Avec le `main()` suivant :

```
public static void main(String[] args) {
    A obj = new A(2);
    A obj2 = obj;
    A obj3 = new A(3);

    if(obj.egal(obj2)) {
        System.out.println("Egal");
    }
    else {
        System.out.println("Different");
    }

    if(obj2.egal(obj3)) {
        System.out.println("Egal");
    }
    else {
        System.out.println("Different");
    }

    if(obj.egal(obj3)) {
        System.out.println("Egal");
    }
    else {
        System.out.println("Different");
    }

    if(obj == obj2){
        System.out.println("Egal");
    }
    else{
        System.out.println("Different");
    }

    if(obj == obj3){
        System.out.println("Egal");
    }
    else{
```

```
        System.out.println("Different");
    }

    if(obj2 == obj3){
        System.out.println("Egal");
    }
    else{
        System.out.println("Different");
    }
}
```

Expliquez la différence entre l'opérateur `==` et la fonction `egal()`.

Exercice 3 *Tableaux d'entiers, tableaux d'objets*

1. Créez un tableau de 100 entiers
2. Peut-on initialiser chaque case de ce tableau avec le mot-clef `null` ? Pourquoi ?
3. La réponse à la question précédente change-t-elle dans le cas d'un tableau d'objets ? Pourquoi ?
4. En considérant les réponses aux questions précédentes, écrivez une classe permettant de ne remplir que n cases d'un tableau à N cases $(N > n)$?

Exercice 4 *Un peu de modélisation* On veut modéliser par un programme Java le fonctionnement d'un parking automobile. Attention : employez judicieusement les mots-clefs `private`,`public` et `static`.

1. Écrivez les classes `Voiture` et `Parking`. Une voiture possède une marque et une couleur et doit pouvoir signaler qu'elle entre dans le parking, ou qu'elle quitte le `Parking`. Un `Parking` doit posséder un nombre de places entier, et doit être capable d'afficher les informations suivantes :
   — Son nombre total de places ;
   — Son nombre de places libres ;
   — La liste des voitures garées dans le parking ;
   — Son pourcentage de remplissage.
2. Améliorez la méthode `entreParking()` de la classe `Voiture` afin qu'une voiture ne puisse pas être dans plus d'un parking à la fois ;
3. Améliorez la classe `Parking` afin que les voitures puissent demander une place précise. La classe `Parking` devra vérifier que cette place est libre. Pensez à utiliser un tableau de `Voiture` !

Exercice 5 *Une classe Polygone* Dans cet exercice, nous allons nous pencher sur la modélisation d'une classe pour gérer les polygones réguliers.

1. Quels champs doit contenir la classe `Polygone` ?

2. Proposez deux méthodes renvoyant l'aire et le périmètre d'un `Polygone`.
3. Proposez une méthode `toString()` qui **affiche** le nombre de côtés d'un `Polygone`.
4. Écrivez une fonction qui affiche un texte définissant les polygones. Cette fonction devrait-elle être `static` ?