

TP #1 Langage Java / Corrigé

Applications Réparties 2009-2010
Université Paris VIII / Parcours SRM / M1

Mise en place de l'environnement de développement :

Un des objectifs de ce module est de vous former à des technologies utilisées couramment dans le monde de l'entreprise. Dans cette optique, bien que simples, les TPs seront réalisés dans un environnement de développement intégré (ou IDE) qui permet de masquer au développeur une partie non-négligeable de la complexité du développement, de la compilation et de l'exécution d'applications.

Un des IDEs les plus utilisés à l'heure actuelle dans le monde Java s'appelle Eclipse, il s'agit d'un projet Open Source soutenu activement par IBM. Il est multi-plateforme et supporte aussi bien les plateformes Windows que Linux ou MacOS.

Téléchargez eclipse à l'adresse suivante : www.eclipse.org et installez le.

Dans eclipse :

1. Définir un nouveau projet Java (File->New->Java Project) et lui donner le nom "tp1".
2. Pour chaque exercice, définir un package tp1.ex# où # correspond au numéro de l'exercice (click droit sur le nom du projet->New->Package). Le code lié à chaque exercice devra se trouver dans un package (espace de nommage) distinct.
3. Si un exercice fait référence à du code qui aurait déjà été implémenté dans un exercice précédent, il est possible d'importer les sources contenues dans ce package de manière à ne pas devoir tout réécrire.

Exercice 1 : Concepts de la Programmation Orientée Objets

Complétez les blancs dans les affirmations suivantes :

1. Les objets du monde réel contiennent des **états** et **comportements**.
2. L'état d'un objet logiciel est stocké dans **ses champs**.
3. Une fabrique d'objets logiciel est appelée une **classe**.
4. Le comportement d'un objet est exposé au travers de **ses méthodes**.
5. Le fait de cacher les données privées d'un objet et de permettre leur accès uniquement via ses méthodes publiques s'appelle **l'encapsulation** des données.
6. Un comportement de haut niveau peut être défini dans **une super-classe ou classe mère** et hérité dans **une classe fille** en utilisant le mot clé **extends**.
7. Un regroupement de méthodes sans implémentation est appelé **une interface**.

Exercice 2 : Classes et Interfaces

Etablissez une interface 'Car' représentant une voiture du monde réel, prévoyez des méthodes pour changer et accéder à la couleur, au nombre de roues, à la vitesse courante du boîtier de vitesse ("gear"), à la vitesse courante de déplacement ("speed").

Correction :

```
public interface Car {
    String getColor();
    void setColor(String newColor);

    int getWheelCount();
    void setWheelCount(int newWheelCount);

    int getCurrentGear();
    void changeGear(int newGear);

    int getCurrentSpeed();
    void speedUp(int increment);
    void speedDown(int decrement);
}
```

Ecrivez ensuite la classe 'BasicCar' implémentant cette interface : vous devez définir les champs de la classe, les initialiser avec des valeurs par défaut et écrire le code des méthodes de l'interface.

Correction :

```
public class BasicCar implements Car {
```

```

private int gear = 0;
private String color = "White";
private int speed = 0;
private int wheelCount = 4;

public void changeGear(int newGear) {
    gear = newGear;
}

public String getColor() {
    return color;
}

public int getCurrentGear() {
    return gear;
}

public int getCurrentSpeed() {
    return speed;
}

public int getWheelCount() {
    return wheelCount;
}

public void setColor(String newColor) {
    if(newColor != null)
        color = newColor;
}

public void setWheelCount(int newWheelCount) {
    if(wheelCount >= 0)
        wheelCount = newWheelCount;
}

public void speedUp(int increment) {
    if(increment > 0)
        speed = speed + increment;
}

public void speedDown(int decrement) {
    if(decrement > 0 && speed - decrement >= 0)
        speed = speed - decrement;
}
}

```

Ecrivez ensuite la classe 'BasicTruck' (camion), sous-classe de la classe 'BasicCar', rajoutant une caractéristique par rapport à une voiture : on est capable de spécifier son type de chargement sous la forme d'une chaîne de caractères et d'accéder à ce type (ex. vide, cailloux, béton, ...).

Correction :

```

public class BasicTruck extends BasicCar {
    String load = "empty";

    public void changeLoad(String newLoad) {
        load = newLoad;
    }

    public String getLoad() {
        return load;
    }
}

```

Quand ces 2 classes et 1 interface sont finalisées, écrivez une classe contenant une méthode main (comme vue en cours) pour créer des instances de ces 2 classes.

Exercice 3 : Opérateurs

Modifiez le programme suivant pour qu'il utilise des opérateurs arithmétiques composés :

```

public class ArithmeticDemo {

    public static void main(String[] args) {

        int result = 1 + 2; // result is now 3
        System.out.println(result);

        result = result - 1; // result is now 2
        System.out.println(result);

        result = result * 2; // result is now 4
        System.out.println(result);

        result = result / 2; // result is now 2
        System.out.println(result);

        result = result + 8; // result is now 10
        result = result % 7; // result is now 3
        System.out.println(result);

    }
}

```

Correction : Voici une solution,

```

class ArithmeticDemo {

    public static void main (String[] args){
        int result = 3;
        System.out.println(result);

        result -= 1; // result is now 2
        System.out.println(result);

        result *= 2; // result is now 4
        System.out.println(result);

        result /= 2; // result is now 2
        System.out.println(result);

        result += 8; // result is now 10
        result %= 7; // result is now 3
        System.out.println(result);

    }
}

```

Ensuite, expliquez pourquoi dans le programme suivant la valeur 6 est affichée deux fois de suite.

```

public class PrePostDemo {
    public static void main(String[] args) {
        int i = 3;
        i++;
        System.out.println(i);
        ++i;
        System.out.println(i);
        System.out.println(++i);
        System.out.println(i++);
        System.out.println(i);
    }
}

```

Correction :

La ligne `System.out.println(++i);` est évaluée à 6 car l'opérateur ++ préfixe incrémente la variable i et retourne la valeur incrémentée (6).

Dans la ligne suivante, `System.out.println(i++);`, l'opérateur ++ postfixe incrémente la variable i à 7 mais uniquement après que la valeur de la variable ai été retournée (encore 6).

Exercice 4 : Expressions, instructions et blocs

Identifiez les expressions suivantes :

