TD 1 - Les objets, les variables

Objectif(s)

- ✯ Pour cette première feuille, je voudrais vous faire penser dans une manière OO.
- ✯ Vous pouvez utiliser le langage algorithmique ou Java, si vous vous sentez déjà suffisamment fort.
- ✯ Vous allez aussi commencer à interpréter du code Java.

Modélisation orientée objet

Exercice 1 – Modélisation dépend du contexte

Supposez que vous voulez définir un schéma pour décrire des mammifères.

Question 1

Quels attributs (d'objet) seraient inclus si ce schéma était défini pour

1. Un éleveur race, taille, poids, prix
2. Un scientifique race, age, sexe
3. Un proprietaire d'un animal de compagnie age, sexe, nom

Question 2

Pour chaque schéma, quels attributs seraient

— variables ? sexe, race, nom

— constants ? age, taille, poids, prix

Exercice 2 – Attribut vs méthode d'objet

Nous revenons à la classe `Cercle` que nous avons vue pendant le cours.
Il y a (au moins) deux possibilités pour définir les propriétés `aire` et `circonference` :

1. Explicitement, en définissant deux attributs.
2. Implicitement, en les dérivant toujours à partir du rayon.

Question 1

Quels sont les avantages/les inconvénients de chaque solution ?

la solution 1 demande une initialisation plus lourde mais moins de calcul et inversement pour la solution 2

Question 2

Si nous avons choisi

1. la version explicite, avons-nous besoin des méthodes "setter", c.-à-d. des méthodes qui mettent à jour les attributs `aire` et `circonference` ?
   — Si oui, comment devraient-elles être implémentées ?
   — Sinon, pourquoi pas ?
2. la version implicite, comment les méthodes "getter", c.-à-d. les méthodes qui récupèrent les valeurs devraient être implémentées ?

Question 3

Étant donné la version explicit, donnez les instructions de la méthode `public void setRayon(double rayon)`.

Code Java

Exercice 3 – Lire et comprendre du code

Nous considérons les deux classes suivantes :

```
class LireCode{
        private int i;
        public boolean r;
        public static final int reponse = 42;

        public void setI(int i){
                this.i = i;
        }

        public int getI(){
                return this.i;
        }

        public static String question1(){
                return "What do you get";
        }

        public static String question2(){
                return "if you multiply six by nine?";
        }
}

public class TestLireCode{
        public static void main(String args[]){
                LireCode e;     initialise un objet LireCode
                LireCode e2 = new LireCode();
                int j;  initialise une variable de type int j

                e2.setI(5);     utilise le setter de l'objet e2
                j = e2.getI();  attribut à j la valeur d'un attribut de e2
                System.out.println(LireCode.reponse); affiche la réponse dans le terminal

                e = e2;   attribut e2 à e
                e.r = true;   attribut la valeur true à l'attribut e.r
                System.out.println(e2.r+" == "+e.r);

                System.out.println();

                e.setI(e2.getI() + 2);
                System.out.println(e2.getI()+" + "+e.getI());

                System.out.println(LireCode.question1()+"
                  ↪ "+LireCode.question2());
```

```
        }
}
```

Question 1
Qu'est-ce qui se passe dans chaque ligne de la méthode `main` de la classe `TestLireCode` ?

Question 2
Quelle est la valeur de j dans

1. la troisième ligne aleatoire
2. la cinquième ligne 5

Question 3
Quelle est la valeur de e dans

1. la première ligne aleatoire
2. la septième ligne e2

Question 4
Le programme affiche quoi ?

42
true == true

5 + 7
What do you get if you multiply six by nine?

Exercice 4 – Les types de variable et leurs visibilités

Question 1

1. Pour chaque paire de variables suivante, quelle est leur différence ?
2. Avons-nous *besoin* d'un accesseur (getter) ou d'un mutateur (setter) pour ces variables ?
3. Pour chaque variable, devrait-on changer leur visibilité ? Pourquoi ?

```
public class Exemple{
 public int var1;
 public final boolean var2;
}
```

final = constante

```
public class Exemple{
 private int var1;
 public static double var2;
}
```

```
public class Exemple{
 private static final int var1;
 private static final char var2 = 'z';
}
```

```
public class Exemple{
 private double [] var1;
 public void oneByOne(){
    int i;
    for(i = 0; i < var1.length; i++){
        System.out.println("Double "+i+": "+var1[i]);
    }
}
```

Question 2

Considerez le code suivant.

```
public class Autre {
  private int x = 11;
  public int y = 13;
  public static int z = 15;
  private static int t = 20;

}
public class Test {
  int a = 3;
  private int b = 7;
  String dix = "dix";
  static boolean vrai = true;

  private void test(int c) {
    int dix = 10:
    for (int i = 0; i<c; ++i) {
      int d = a + b;
    }
  }

  public Test() {
    Autre autre = new Autre();
    test(42):
  }
}
```

1. Quelles sont les variables accessibles dans la méthode `test()` et quelle est leur valeur ?
2. Quelles sont les variables accessibles dans le constructeur `Test()` et quelle est leur valeur ?