Feuille de TD/TP n° 3

Classes et objets - 2 : constructeurs et surcharge

Exercice 1 : Qu'affiche le programme suivant ?

```
class Surcharge {
    int a = 0;

    public int f(int n, int m) {
        a = n + m;
        return a;
    }
    public String f(int n, String m) {
        a += n;
        return "(" + String.valueOf(a) + ", " + m + ")";
    }
    public String g(int n) {
        a = n;
        return String.valueOf(a);
    }
    public int g(String n) {
        a++;
        System.out.print(n);
        return a;
    }
    public static void main(String[] args) {
        Surcharge s = new Surcharge();
        System.out.println(s.f(s.g(s.g(s.g("A"))),s.g(s.f(5,s.g("B")))));
    s}
}
```

Exercice 2 : Qu'affiche le programme suivant ?

```
class Constructeur {
    int x, y;

    Constructeur(int x, int y) {
        this.x = x;
        this.y = this.x + y;
    }
    Constructeur(String s) {
        this();
        System.out.print(s);
        x = y * 2 + x;
    }
    Constructeur(int a) {
```

```
        this(5, a);
        System.out.print("=>");
        x = x + a;
        y++;
    }
    Constructeur() {
        this(4);
        y = x + y;
        x--;
    }

    public static void main(String[] args) {
        Constructeur c = new Constructeur("Hello !");
        System.out.println("(" + c.x + ", " + c.y + ")");
    }
}
```

Exercice 3 : Modifier la classe `Rationnel` de la feuille n°2 afin :

- de créer un rationnel avec la valeur 0 par défaut,
- de créer un rationnel dont la valeur est un entier passé en paramètre,
- de créer un rationel à partir de la donnée des valeurs du numérateur et du dénominateur.

Peut-on ajouter les méthodes suivantes ? Justifiez et ajoutez-les si c'est possible.

- `void ajouter(int n)` : ajoute la valeur `n` au rationnel.
- `void multiplier(Rationnel r)` : multiplie le rationnel par `r` et le remplace par le résultat.
- `void multiplier(int n)` : multiplie le rationnel par `n` et le remplace par le résultat.
- `Rationnel multiplier(int n)` : multiplie le rationnel par `n` et renvoie le résultat.

Exercice 4 : Écrire une classe `PaireEntier` implémentant une paire d'entier :

1. Définir un constructeur qui initialise les attributs de la paire. Définir une méthode `affiche` et une fonction `main` pour tester cette classe.
2. Définir un deuxième constructeur, qui initialisera à 0 les composants de la paire.
3. Définir un troisième constructeur, qui initialisera une paire à l'aide d'une autre paire.
4. Définir des méthodes permettant d'accéder et de modifier chaque élément de la paire.

Définir une nouvelle classe `RationnelPaire` qui possède les mêmes fonctionnalités que la classe `Rationnel` mais qui utilise un objet de la classe `PaireEntier` pour représenter le rationnel.