TP 2

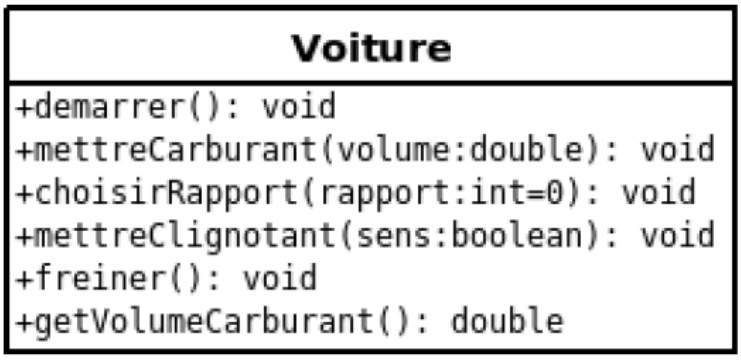
Utilisation d'objets & Expression en Java

Objectif : les objectifs de ce TP sont :

- Première appropriation du modèle objet. Distinguer action et propriété.
- commencer à faire la liaison entre le concept général d'objet et le langage Java....

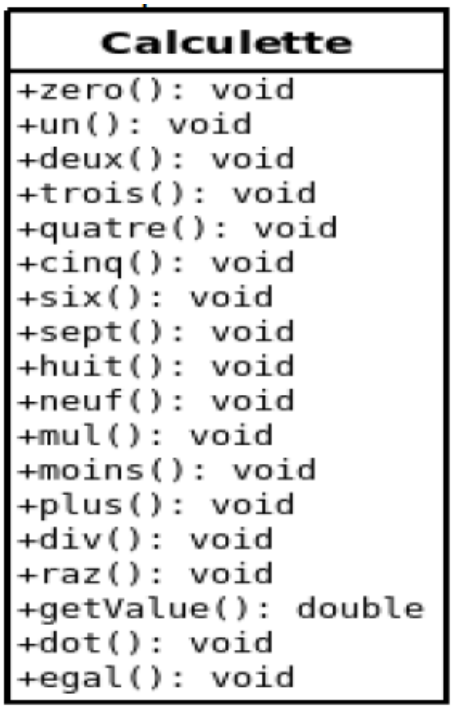
Exercice 1

On considère la classe définie en colonne de gauche. Traduire en une suite d'instructions d'un pseudo langage objet la série des actions de la colonne de droite:



- Je démarre le moteur de la voiture et passe la première.
- Je passe la seconde et mets le clignotant à gauche.
- Je consulte la jauge à essence.
- Je freine, met le clignotant à droite, et coupe le moteur.
- Dans une station-service je mets 52.5 l de carburant

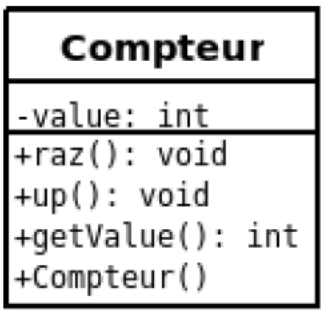
Même question :



- Je réalise avec la calculette ci-dessus l'opération suivante : 564 / 2 = ...
- Je consulte le résultat
- Je remets la calculette à zéro.

Exercice 2

On considère la classe Compteur ou sont explicites le constructeur (Compteur) et l'attribut interne de l'objet (value). Traduire les actions ci-dessous en instructions appliquées a ce compteur. On utilisera si possible Java à la place du pseudo langage utilisé en exercice 1. Le signe **-** place devant value signifie que cet attribut est prive et ne peut en conséquence pas être accédé de l'extérieur de l'objet. Les signes **+** signifient que les entités désignées sont utilisables sans restriction.



- On crée un nouveau compteur;
- On effectue une remise à zéro sur ce compteur.
- On incrémente trois fois ce compteur
- On consulte sa valeur.
- On incrémente encore une fois
- On consulte la valeur.
- On effectue une remise à zéro sur ce compteur.
- On consulte la valeur.

Exercice 3

En vous aidant du cours implémenter en Java un tel Compteur. Le tester en appliquant la série d'instructions évoquées dans la question précédente.
Remarque : la phrase « on consulte sa valeur » peut par exemple être implémentée en provoquant un affichage de la valeur en question a l'écran (par **System.out.println(...)** en java)

Exercice 4

Reformuler la classe Compteur en explicitant le plus possible l'emploi du mot réservé **this**.

Exercice 5

Définissez une classe Complexe, pour représenter les nombres de l'ensemble **C**. Un objet complexe aura deux attributs, une partie réelle et une partie imaginaire : **a+ib**.

Vous définirez un constructeur par défaut qui initialisera les deux attributs à zéro, ainsi qu'un constructeur qui initialisera un nombre complexe à partir de deux paramètres réels.

Écrire une méthode **toString()**. La méthode **toString** permet la conversion d'un objet de type complexe en une chaîne de caractères. Elle sera utilisée implicitement par certaines méthodes, par exemple **System.out.println**. Ainsi l'appel à la méthode **System.out.println(c)** ; avec c de type complexe utilisera la méthode **toString()** de la classe Complexe.

Complétez la classe Complexe avec les opérations d'addition et de multiplication. Testez ces deux méthodes.

Exercice 6

Nous allons dans cet exercice reprendre l'exemple sur les équations du second degré et le coder en utilisant les principes de la programmation orientée objet (en utilisant des classes et des méthodes).
Le but est de résoudre une équation de la forme : $ax^2 + bx + c = 0$
Définir une classe **EqSecondDegre** avec les caractéristiques suivantes :

- la classe possède les attributs réels suivants : r1 et r2 (les éventuelles solutions), delta (le discriminant), a, b et c les coefficients du polynôme;
- écrire un constructeur **EqSecondDegre** à trois paramètres réels x1,x2 et x3 qui correspondent aux coefficients du polynôme à résoudre. Ce constructeur affectera les valeurs passées en paramètre aux attributs a, b et c et calculera la valeur du discriminant delta;
- écrire une méthode **afficheDiscriminant()** qui affiche la valeur du discriminant;
- écrire une méthode résoudre() qui résout l'équation et affecte aux attributs r1 et r2 les racines des solutions (on ne prendra pas en compte le cas des solutions complexes);
- écrire une méthode **afficheSolutions()** qui affiche les solutions de l'équation.

Utiliser le programme Test.java suivant afin de tester votre classe :

```
public class Test {
	// Résolution de l'équation -2x² + x + 3
	public static void main(String[] args) {
		EqSecondDegre equation = new EqSecondDegre(-2.0, 1.0, +3.0);
		equation.afficheDiscriminant();
		equation.resoudre();
		equation.afficheSolutions();
	}
}
```

Vous devriez obtenir quelque chose qui ressemble à la sortie écran ci-dessous :

25.0
La première racine est 1.5
La deuxième racine est -1.0

Comme vous avez pu le constater dans notre programme Test.java il est nécessaire d'appeler la Méthode résoudre() avant la méthode afficheSolutions().

- modifier votre classe afin que la méthode afficheSolutions() appelle la méthode résoudre (), celle-ci ne pourra être appelée à l'extérieur de la classe;
- modifier votre classe afin qu'elle affiche qu'il n'y a aucune solution si le discriminant est nul.

Exercice 7

Utilisez la classe Complexe de l'exercice 5 afin de représenter les racines de l'équation du second degré si la valeur du discriminant est négative.

On modifie la classe Test pour avoir un discriminant négatif

```
public class Test {
	// Résolution de l'équation 2x2 + x + 3
	public static void main(String[] args) {
		EqSecondDegreC equation = new EqSecondDegreC(2.0, 1.0, +3.0);
		equation.afficheDiscriminant();
		equation.resoudre();
		equation.afficheSolutions();
	}
}
```

Résultat :

```
-23.0
(-0.25 +i*1.1989578808281798)
(0.25 +i*1.1989578808281798)
```