JAVA – HOWTO 2
Mener à bien des calculs

Université de Mons-Hainaut

Beaucoup de problèmes de programmation nécessitent l'utilisation de formules pour calculer des valeurs. Il n'est pas toujours trivial de savoir comment transformer l'énoncé d'un problème en une suite de formules mathématiques à calculer, ni en dernier ressort de les programmer en JAVA.

- **Étape 1** Comprendre le problème, quelles en sont les entrées et les sorties ?

  Prenons un exemple. Supposons que l'on vous demande de simuler le fonctionnement d'une machine qui vend des timbres, selon le procédé suivant : l'utilisateur doit insérer une certaine somme (multiple de 0.10€), et lorsqu'il appuie sur le bouton, la machine lui délivre autant de timbres à 0.40€ que la somme introduite le permet, puis la différence en timbres à 0.10€.

  Dans ce problème, il n'y a qu'une seule entrée :

  – la somme introduite par l'utilisateur

  Et il y a deux sorties à déterminer :

  – le nombre de timbres à 0.40€ délivrés
  – le nombre de timbres à 0.10€ délivrés

- **Étape 2** Traiter des exemples "à la main".

  Étape très importante. Si vous arrivez à le faire, vous serez capable d'automatiser la procédure de calcul. Ici par exemple pour 1€ inséré, la machine délivrera 2 timbres à 0.40€, et 2 timbres à 0.10€. L'état de la machine lors de la transaction, représenté par un "montant", passe par les 4 valeurs suivantes : O€, 1€, 0.20€, 0€.

- **Étape 3** Trouver les équations qui calculent les réponses.

  Il faut transformer en variables les valeurs d'exemples que vous avez pris dans les calculs à la main.

  Ici on peut prendre : I pour la somme insérée, x, y pour les nombres de timbres à 0.40€, et à 0.10€ délivrés. L'équation est :

  $$I = 0.40x + 0.10y, \text{ avec } x \text{ maximal.}$$

  Ainsi x sera la partie entière de $\frac{I}{0.40}$, notation : $x = \lfloor \frac{I}{0.40} \rfloor$
  et $y = (I - 0.40x)/0.10$.

- **Étape 4** Écrire les les équations sous forme de code en Java.

  On se sert des fonctions mathématiques de Java.

  Détailler l'état de la machine par une variable `montant`.

```
montant = I;
x = (int)Math.floor(montant/0.40);
montant = montant - x * 0.40;
y = (int)Math.round(montant/0.10);
montant = 0;
```

  Notez le "cast" : on doit faire précéder les fonctions `Math.floor(..)`, et `Math.round(..)` par `(int)`, car en fait celles-ci renvoient un `double` (cf. l'API Java), et l'on veut des entiers.

- **Étape 5** Construire une classe qui va mener à bien ces calculs.

  On renvoie au HOWTO 1 qui explique comment choisir ses méthodes et classes. Ici on trouve trois méthodes :

  - `void insert(double somme)`
  - `int nombre04()`
  - `int nombre01()`

  L'état de la machine est représenté par une variable d'instance : `montant`. D'où la classe suivante :

```
public class TimbreMachine
{
   public TimbreMachine()
   {
   montant = 0;
   }
   public void insert(double somme)
   {
      montant = montant + somme;
   }
   public int nombre04()
   {
      int x = (int)Math.floor(montant/0.40);
      montant = montant - x * 0.40;
      return x;
   }
   public int nombre01()
   {
      int y = (int)Math.round(montant/0.10);
      montant = 0;
      return y;
   }
   private double montant;
}
```

- **Étape 6** Tester cette classe.

  Vérifier les résultats fournis par les calculs à la main.
  Ici :

```
public class TestMachine
{
  public static void main(String [] args)
  {
   TimbreMachine M = new TimbreMachine;
   M.insert(1);
System.out.println("Nombre de timbres
à 0.40 :" + machine.nombre04());
System.out.println("Nombre de timbres
à 0.10 :" + machine.nombre01());
  }
}
```