TP 5 : fonctions, boucles `while`

Informatique Fondamentale (IF1)

Semaine du 26 Octobre 2008

1 Fonctions

Exercice 1. Sans utiliser les opérateurs logiques `||`, `&&`, `!` etc., écrivez les quatre fonctions suivantes dans une même classe `OperateursBooleens` :

- « `not` » de signature `public static boolean not(boolean b)`, qui prend un booléen en paramètre, et qui calcule sa négation ;
- « `and` » de signature `public static boolean and(boolean b, boolean c)`, qui prend deux booléens en paramètre, et qui calcule leur conjonction ;
- « `or` » de signature `public static boolean or(boolean b, boolean c)`, qui prend deux booléens en paramètre, et qui calcule leur disjonction ; et
- « `xor` » de signature `public static boolean xor(boolean b, boolean c)`, qui prend deux booléens en paramètre, et qui calcule leur disjonction exclusive.

Testez vos quatre fonctions (sur toutes les valeurs possibles de leurs paramètres) dans le programme principal de la classe, en affichant leurs résultats.

Exercice 2. Dans une classe `Trie`, écrivez trois fonctions appelées `premier`, `deuxieme` et `troisieme`, qui prennent chacune en paramètre trois entiers, et qui calculent respectivement le plus grand, le deuxième et le plus petit des trois entiers.

Par exemple, `premier` aura la signature

```
public static int premier(int a, int b, int c).
```

Dans le programme principal de la classe, écrivez un programme qui lit trois entiers sur l'entrée standard et les affiche dans l'ordre croissant. Pour cela, utilisez les fonctions `premier`, `deuxieme` et `troisieme`.

2 Boucles indéfinies

Exercice 3. Écrivez un programme `Encore` qui demande à l'utilisateur « encore ? » , et qui continue de lui poser la question tant que celui-ci lui répond « oui ».

Exercice 4. La commande Unix « `yes` » affiche indéfiniment sur la console des lignes contenant le caractère « `y` ». Écrivez un programme Java « `Yes` » qui a le même comportement.

Vous pouvez interrompre ce programme en tapant `^C` (tenez la touche *Control* enfoncée pendant que vous tapez un c).

Exercice 5. Écrivez un programme qui demande des entiers tant que l'utilisateur rentre un entier positif puis affiche leur somme et leur moyenne ;

3 Des approximations numériques

Exercice 6. Étant donné un réel positif a, on définit la suite réelle $(x_n)_{n\in\mathbf{N}}$ de la manière suivante :

$$\begin{aligned} x_0 &= a \\ x_{i+1} &= \frac{x_i^2 + a}{2x_i} \qquad i > 0 \end{aligned}$$

Cette suite converge vers $\sqrt{a}$.

Écrivez une fonction `mysqrt` de signature

```
public static double mysqrt(double a, int n)
```

qui prend en paramètre un flottant a et un entier n et qui retourne une valeur approchée de $\sqrt{a}$ en utilisant l'approximation x_n.

Écrivez une fonction `main` qui lit un flottant a et un entier n et affiche $b =$ `mysqrt`(a, n). Affichez aussi b^2 pour vérifier.

Exercice 7. Un étudiant place 1 zł[1] dans une banque. Cette somme sera rémunérée au taux de 100% à la fin de l'année[2], l'étudiant se retrouvera donc en possession de 2 zł au bout d'un an. Un deuxième étudiant choisit de placer son zł dans une banque lui offrant un taux de rémunération de 50% tous les six mois. Ce dernier se retrouvera en possession de $2,25$ zł à la fin de l'année.

Écrivez une fonction qui calcule ce que l'étudiant n, qui à placé son złoty dans une banque lui offrant un taux de rémunération de $1/n$ toutes les $1/n$ années, possède à la fin de l'année. Écrivez un programme (une fonction `main`) qui vous permette de tester cette fonction.

Votre programme calcule des valeurs approchées de e. Estimez vous que la convergence est rapide ?

4 Ruptures de boucle

Exercice 8.

1. Écrivez un programme qui affiche la table de multiplication :

```
 1  2  3  4  5 ... 10
 2  4  6  8 10 ... 20
...
10 20 30 40 50 ...100
```

[1] *Złoty*, l'unité monétaire polonaise.

[2] Exceptionnel, ou alors donnez-moi l'adresse de votre banque.

Pour aligner les valeurs, vous pourrez utiliser une expression de la forme

```
(n >= 10 ? "" : " ") + n
```

2. Écrivez un programme qui lit un entier n puis qui affiche la table de multiplication en s'arrêtant dès qu'il rencontre n.

```
12
1  2  3  4  5 ... 10
2  4  6  8 12
```

3. Écrivez une fonction qui prend un entier en paramètre et retourne `true` si et seulement si cet entier apparaît dans la table de multiplication, en prenant soin de cesser de parcourir la table dès lors que la valeur est trouvée. Écrivez une fonction `main` et testez votre programme.

5 Algorithme d'Euclide

Exercice 9. L'algorithme dit d'Euclide permet de calculer le pgcd de deux entiers positifs non nuls. Il est basé sur les propriétés suivantes :

- si $a < b$, alors $\text{pgcd}(a, b) = \text{pgcd}(b, a)$;
- si b divise a, alors $\text{pgcd}(a, b) = b$;
- sinon, $\text{pgcd}(a, b) = \text{pgcd}(b, r)$, où r est le reste de la division euclidienne de a par b.

Écrivez une fonction `pgcd` qui calcule le pgcd de ses deux paramètres à l'aide de l'algorithme d'Euclide. Comme d'habitude, écrivez une fonction `main` qui vous permette de la tester.