Université de Nice – Sophia Antipolis — Deug MASS MIAS SM 1re année
UEF1 : Informatique & Programmation — 2000–2001

TP N° 7

Premières itérations

Ça va mieux en le révisant ... Une *itération* est le fait de répéter l'exécution d'un ensemble d'instructions : on parle de *boucle*. Les deux instructions composées `while` et `do...while` permettent d'écrire de telles boucles. A priori, on ne sait pas combien de fois on va répéter une boucle. Toutefois, pour ne pas répéter indéfiniment un bloc d'instructions, on a besoin de limiter la répétition à l'aide d'une condition. Une condition, c'est une expression booléenne : si elle est évaluée à `true` elle permet de continuer à boucler, si elle est évaluée à `false` elle met fin à la répétition.
Les instructions `while` et `do...while` se distinguent par l'emplacement de leur condition. Elle est évaluée d'emblée dans le `while` alors qu'elle n'est évaluée qu'à l'issue d'un premier passage dans le bloc d'instructions dans le `do...while`.
Notez qu'au moins une des variables qui apparaissent dans la condition doit être modifiée dans le bloc d'instructions à répéter[1]. Il faut bien que le résultat de l'évaluation ait une chance d'être modifié pour que la répétition puisse s'arrêter.

1 Petites itérations

Exercice 1)

a. Ecrivez un programme qui demande à l'utilisateur de saisir son nom puis de choisir un entier n compris entre 10 et 20. Le programme devra alors afficher n fois le nom de l'utilisateur, en passant à la ligne à chaque fois.
b. Améliorez le programme précédent afin de prévoir le cas où l'utilisateur donnerait un entier non compris entre 10 et 20. Ce programme devra persister à demander un entier dans l'intervalle voulu jusqu'à ce que l'entier lu soit correct. ◇

Exercice 2) Devinette On se propose d'écrire un programme où l'ordinateur choisit un nombre au hasard que l'utilisateur devra deviner.

a. Commencez par faire choisir un entier entre 1 et 100 à l'ordinateur. Demandez à l'utilisateur de deviner ce nombre et recueillez sa réponse. Il devra rééssayer jusqu'à obtention de la bonne réponse. De plus, après chaque essai infructueux, l'ordinateur devra en outre préciser *Trop grand* ou bien *Trop petit* afin de hâter la victoire de l'utilisateur.
Conseil Si cela peut vous aider, schématisez ce programme à l'aide d'un ordinogramme.
b. Améliorez ce programme de sorte qu'à la fin, il indique à l'utilisateur le nombre d'essais qu'il lui a fallu pour deviner le nombre en question. ◇

1. Vous verrez plus tard la notion de boucle infinie, qui dépasse du cadre de notre enseignement.

Exercice 3) Ecrivez un programme de conversion Francs/Euros. Ce programme devra demander à la fin de chaque boucle à l'utilisateur s'il veut continuer ou pas. Que la réponse de l'utilisateur soit `Non, non, oui, Oui` ou même `N, n, O, o`, le programme devra respecter le choix de l'utilisateur. Voici un petit squelette du programme attendu :

```
...
boolean fini;
do
{  ...
   String reponse = readLine("Voulez-vous continuer ?");
   fini = reponse.substring(0,1).toLowerCase().equals("n");
}
while (! fini)
...
```

On aurait pu se servir de la méthode `readBoolean`(...) de la classe `Console` pour lire la réponse de l'utilisateur directement sous forme booléenne. Ecrivez une variante de cette classe qui utilise l'idée précédente. ◇

2 Une méthode itérative

Exercice 4) Factorielle itérative

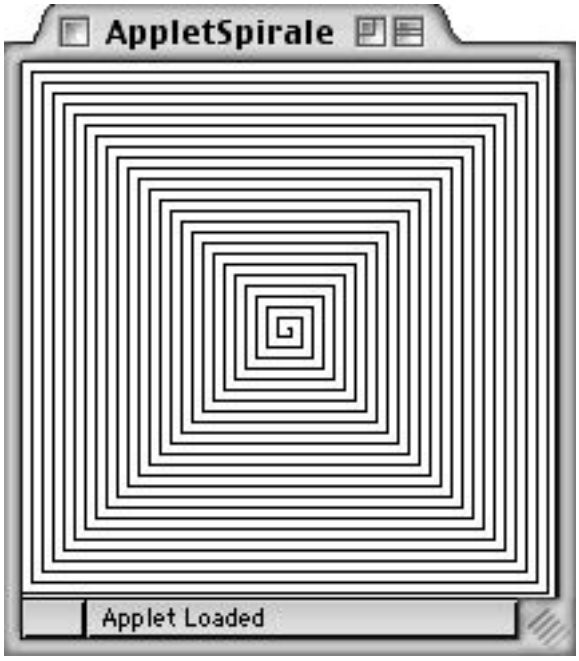
1. Ajoutez à votre classe `Numerik` une méthode statique dont l'entête est le suivant :
   ```
   static int factorielle (int n)
   ```
   Cette méthode doit renvoyer la factorielle de l'entier passé en paramètres, i.e. le produit des n premiers entiers dès que $n > 0$. On rappelle à toutes fins utiles que $0! = 1$.
2. Vous chercherez par tâtonnements le plus grand entier n dont cette méthode est capable de calculer la factorielle. ◇

3 Approfondissement

Programmez une *applet* de taille 200×200 dans laquelle vous dessinerez une *spirale à coins carrés* comme dans la copie d'écran ci-dessous. Vous utiliserez une boucle `do ... while (...)` qui terminera dès que le tracé commence à sortir de la fenêtre. L'instruction principale à itérer est la méthode d'instance `drawLine`(...) de la classe `Graphics`, qui trace un segment de droite entre deux points.

Déclarez en constante la dimension de l'*applet*, de manière à pouvoir la recompiler rapidement en

ne modifiant qu’un seul endroit du texte-source…



◇