TP 6 : boucles indéfinies

Introduction à l'informatique et à la programmation (IF1)

Chaque fonction demandée est évidemment à écrire dans une classe et à tester avec un `main`, même si ce n'est pas écrit de façon explicite dans l'énoncé.

1 Boucles indéfinies

▶ **Exercice 1** : Écrivez une fonction `encore` qui demande à l'utilisateur « encore ? » , et qui continue de lui poser la question tant que celui-ci lui répond « oui ».

▶ **Exercice 2** : La commande Unix `yes` affiche indéfiniment sur la console des lignes contenant le caractère `y`. Écrivez un programme Java `Yes.java` 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 3** : Écrivez une fonction qui lit des entiers jusqu'à ce que l'utilisateur entre 0, puis qui affiche la somme des entiers entrés par l'utilisateur.
Modifiez votre fonction pour qu'elle affiche la somme et le produit des entiers entrés par l'utilisateur.

2 Des approximations numériques

▶ **Exercice 4** (Approximation de la racine carrée — méthode de Héron) : Étant donné un réel strictement positif a, on définit la suite réelle $(x_n)_{n\in\mathbf{N}}$ de la manière suivante :

$$\begin{array}{rcll} x_0 & = & a & \\ x_{i+1} & = & \dfrac{x_i + \frac{a}{x_i}}{2} & \quad i > 0 \end{array}$$

Cette suite converge vers $\sqrt{a}$.
Écrivez une fonction `mysqrt` qui prend en paramètre un réel a et un entier n et qui retourne une valeur approchée de $\sqrt{a}$ en utilisant l'approximation x_n.
Vérifiez le comportement de votre fonction `mysqrt` en remettant au carré le résultat obtenu (dans un `main`).

▶ **Exercice 5** (Approximation de e) : Un étudiant place 1€ dans une banque. Cette somme sera rémunérée au taux de 100%, l'étudiant se retrouvera donc en possession de 2€ au bout d'une année. Un deuxième étudiant choisit de placer son euro dans une banque lui offrant un taux de rémunération de 50% tous les six mois. Ce dernier se retrouvera en possession de 2,25€ à la fin de l'année.

Écrivez une fonction `pecule` qui calcule ce que l'étudiant n, qui à placé son euro 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 qui vous permette de tester cette fonction.
Votre programme calcule des valeurs approchées de e. Estimez-vous que la convergence est rapide ?

▶ **Exercice 6** (Approximation de p) : On part de l'égalité suivante (qu'on ne vous demande pas de montrer) :

$$\sum_{n=1}^{\infty} \frac{1}{n^2} = \frac{p^2}{6}$$

pour calculer une valeur approchée de p. Pour cela, on définit les suites $(S_N)_N$ et $(p_N)_N$ de la façon suivante :

$$S_N = \sum_{n=1}^{N} \frac{1}{n^2} = \frac{p_N^2}{6},$$

où la limite de $(p_N)_N$ est p. La différence $S_{N+1} - S_N$ tend vers 0 et c'est grâce à cette différence qu'on va évaluer si on est suffisamment proche de la valeur recherchée ou non, en utilisant le fait que

$$S_{N+1} - S_N = \frac{1}{6}\ p_{N+1} - p_N \quad p_{N+1} + p_N\ .$$

Donc pour N suffisament grand

$$S_{N+1} - S_N > p_{N+1} - p_N.$$

Écrivez une fonction qui prend en argument un réel représentant une erreur admissible pour la différence de deux éléments consécutifs de la suite $(p_N)_N$ et renvoie une valeur approchée de p.

3 Algorithme d'Euclide

▶ **Exercice 7** (Algorithme d'Euclide) : L'algorithme dit d'Euclide permet de calculer le pgcd de deux entiers strictement positifs a et b.
L'algorithme manipule deux entiers strictement positifs $a \geq b$. Initialement, a vaut $\max(a,b)$, et b vaut $\min(a,b)$.
À chaque étape, on calcule le reste r de la division de a par b. Si ce reste est nul, alors l'algorithme termine, et le résultat est b. Sinon, a prend l'ancienne valeur de b, et b prend la valeur de r.
Écrivez une fonction *non récursive* `pgcd` qui calcule le pgcd de ses deux paramètres entiers à l'aide de l'algorithme d'Euclide.

4 Pour ceux qui ont le temps

▶ **Exercice 8** (Approximation de p) : Cherchez et implémentez d'autres méthodes d'approximation de p que celle proposée à l'exercice 6.