

feuille de travaux pratiques

Séance 3 : boucles et récursivité

Le symbole $\diamond$ indique un exercice optionnel. Le travail demandé peut être effectué indifféremment avec MATLAB ou le logiciel libre GNU OCTAVE (<http://www.gnu.org/software/octave/>).

Exercice 1 (suite de Fibonacci).

1. Écrire une boucle calculant les valeurs des vingt premiers termes de la suite de Fibonacci définie par

$$u^{(0)} = 0, u^{(1)} = 1 \text{ et } u^{(k+2)} = u^{(k+1)} + u^{(k)}, \forall k \in \mathbb{N},$$

et conservant ces valeurs dans un tableau.

2. Écrire une boucle calculant les termes successifs de la suite de Fibonacci dont la valeur est inférieure ou égale à 50000 et afficher le dernier de ces termes.
3. Écrire enfin une fonction `fibonacci(n)` calculant de manière itérative le $n^{\text{ième}}$ terme de la suite de Fibonacci, sans toutefois conserver les valeurs de tous les termes de la suite.

Exercice 2 (suites adjacentes). On définit deux suites $(u^{(k)})_{k \in \mathbb{N}}$ et $(v^{(k)})_{k \in \mathbb{N}}$ par

$$u^{(0)} = 1, v^{(0)} = 2, u^{(k+1)} = \frac{u^{(k)} + v^{(k)}}{2}, v^{(k+1)} = \sqrt{u^{(k+1)}v^{(k)}}, \forall n \in \mathbb{N}.$$

On admet que ces suites sont adjacentes, de limite $\frac{\sqrt{27}}{\pi}$.

1. Écrire un programme lisant ¹ un entier n et affichant l'approximation du nombre π obtenue à partir de la valeur de $v^{(n)}$.
2. Écrire un programme lisant un réel ε strictement positif et affichant l'approximation du nombre π obtenue à partir de la valeur de $v^{(n)}$, premier terme de la suite $(v^{(k)})_{k \in \mathbb{N}}$ à satisfaire la condition

$$\left| \frac{u^{(n)} - v^{(n)}}{u^{(n)} + v^{(n)}} \right| \leq \varepsilon,$$

avec ε un réel strictement positif fixé

Exercice 3 $\diamond$ (développements en série entière de cos et sin). Écrire une fonction `cosn(n,x)`, prenant comme arguments un entier naturel non nul n et un réel x , calculant une approximation de la valeur de la fonction cosinus en x obtenue en ne conservant que les n premiers termes du développement en série entière

$$\cos(x) = 1 - \frac{x^2}{2!} + \dots + (-1)^k \frac{x^{2k}}{(2k)!} + \dots \quad (1)$$

Comparer les résultats de cette fonction avec ceux de la commande `cos(x)` pour différentes valeurs de n et de x . Écrire de la même manière une fonction `sinn(n,x)` basée sur le développement en série entière de la fonction sinus

$$\sin(x) = x - \frac{x^3}{3!} + \dots + (-1)^k \frac{x^{2k+1}}{(2k+1)!} + \dots$$

1. Utiliser pour cela la fonction `input`.

Exercice 4 (programmation récursive). En informatique, une fonction dite récursive lorsqu'elle s'appelle elle-même. En pratique, une telle fonction aura toujours au moins une instruction conditionnelle, afin que, dans certains cas au moins, il n'y ait pas d'appel récursif (sans quoi la fonction s'appellerait indéfiniment jusqu'à la saturation de la pile, provoquant une interruption du programme). Le concept de fonction récursive est généralement opposé à celui de fonction itérative, qui s'exécute sans s'invoquer ou s'appeler explicitement.

Bien que cette forme de programmation aboutisse à des programmes concis et proches des formulations mathématiques qui en sont à l'origine, il peut parfois être mal indiqué ou même catastrophique d'employer la récursivité (toute fonction récursive pouvant être remplacée par une fonction itérative), comme on le vérifiera à la troisième question du présent exercice.

1. Écrire une fonction récursive `rfactorielle(n)` calculant $n!$.
2. Écrire, en utilisant la fonction `rem` donnant le reste de la division euclidienne de deux entiers, une fonction récursive `rpgcd(a,b)` renvoyant le plus grand commun diviseur² des entiers naturels a et b calculé par l'algorithme d'Euclide³.
3. Écrire une fonction récursive `r fibonacci(n)` calculant le $n^{\text{ième}}$ terme de la suite de Fibonacci et comparer son temps d'exécution avec celui de la fonction `fibonacci(n)` de l'exercice 1.
4. Écrire une fonction récursive `rcollatz(n)` renvoyant la valeur 1 si la conjecture de Collatz⁴ est vérifiée pour l'entier $n > 0$.
5. Écrire une fonction récursive `rcosn(n,x)` de calcul d'une approximation de $\cos(x)$ vue dans l'exercice 3 utilisant la relation entre les termes de la somme (1), c'est-à-dire

$$u^{(0)} = 1 \text{ et } u^{(k)} = -\frac{x^2}{2k(2k-1)} u^{(k-1)}, \forall k \geq 1.$$

Exercice 5 $\diamond$ (procédé Δ^2 d'Aitken). On peut obtenir une valeur approchée du réel π en sommant un nombre fini de termes de la série de Madhava–Gregory–Leibniz,

$$\sum_{n=0}^{+\infty} \frac{(-1)^n}{2n+1} = \frac{\pi}{4}.$$

La convergence de cette série est malheureusement lente et, pour l'accélérer, on se propose d'utiliser le procédé Δ^2 d'Aitken. Cette technique consiste en la construction d'une suite $(u^{(k)})_{k \in \mathbb{N}}$ définie par

$$u^{(k)} = s^{(k)} - \frac{(s^{(k+1)} - s^{(k)})^2}{s^{(k)} - 2s^{(k+1)} + s^{(k+2)}}, \text{ avec } s^{(m)} = \sum_{n=0}^m \frac{(-1)^n}{2n+1}, m \geq 0,$$

ayant même limite que la suite $(s^{(k)})_{k \in \mathbb{N}}$ des sommes partielles de la série de Madhava–Gregory–Leibniz et convergeant plus rapidement.

1. Écrire une boucle calculant les termes de la suite $(s^{(k)})_{k \in \mathbb{N}}$ et s'arrêtant lorsque la condition $|s^{(k)} - \frac{\pi}{4}| \leq \varepsilon$ est vérifiée, avec ε un réel strictement positif fixé. En prenant $\varepsilon = 10^{-6}$, combien faut-il calculer de termes pour satisfaire le critère?
2. Modifier la boucle de façon à calculer les termes de la suite $(u^{(k)})_{k \in \mathbb{N}}$. Pour quelle valeur de l'entier k a-t-on $|u^{(k)} - \frac{\pi}{4}| \leq \varepsilon$, avec $\varepsilon = 10^{-6}$?
3. Reprendre les questions précédentes avec $\varepsilon = 10^{-8}$.

2. C'est-à-dire le plus grand entier naturel qui divise simultanément ces deux entiers.

3. Cet algorithme est basé sur la propriété suivante : on suppose que $a \geq b$ et on note r le reste de la division euclidienne de a par b ; alors le pgcd de a et b est le pgcd de b et r . En pratique, il suffit donc de faire des divisions euclidiennes successives jusqu'à trouver un reste nul.

4. On appelle suite de Syracuse toute suite d'entiers naturels définie de la manière suivante : on part d'un nombre entier plus grand que zéro ; s'il est pair, on le divise par deux ; s'il est impair, on le multiplie par trois et on ajoute un au résultat, la suite étant obtenue en répétant cette opération. Après que le nombre 1 a été atteint, la suite devient périodique, les valeurs (1, 4, 2) se répétant indéfiniment en un cycle appelé cycle trivial. La conjecture de Collatz affirme que les suites de Syracuse de tous les nombres entiers strictement positifs atteignent le cycle trivial.