TRAVAUX PRATIQUES MAPLE NO. 3 (DEUXIÈME TRIMESTRE)

CPES FEYDER II

Suites récurrentes

Exercice 1 : Algorithme des Babyloniens. Voici un algorithme très simple, déjà utilisé du temps de Babylone, pour calculer à la main une valeur approchée de la racine carrée d'un nombre. Supposons que l'on cherche à calculer la racine carrée $\sqrt{a}$ du nombre a. On considère la suite récurrente suivante :

$$u_{n+1} = \frac{1}{2}(u_n + \frac{a}{u_n}),\ u_0 = 1\ .$$

Pour n petit, de l'ordre de 5, u_n est déjà une très bonne approximation de $\sqrt{a}$, avec par exemple une vingtaine de décimales exactes pour $a = 2$. Et le nombre de décimales exactes double à chaque itération !

(1) Écrire une procédure `baby` qui prend comme arguments le nombre a et l'entier n, et qui renvoie la valeur de u_n ;

(2) Utiliser cette procédure pour calculer $\sqrt{2}, \sqrt{3}, \sqrt{4}$ et $\sqrt{5}$ avec successivement $n = 2, 3, 4$ et 5. Comparer, pour chaque exemple, le résultat obtenu avec la valeur approchée, à 30 décimales près, calculée par la commande `evalf(...,30)`. Indiquer le nombre de décimales exactes.

Exercice 2 : Polynômes de Tchebichev. Les polynômes de Tchebichev $T_n(\mathtt{x})$ sont définis par la propriété $(\mathcal{P})$:

$$T_n(\cos a) = \cos na,\ \forall a \in \mathbb{R}\ .$$

En outre, on peut les obtenir par la récurrence d'ordre 2 suivante :

$$T_{n+2}(\mathtt{x}) = 2\mathtt{x}T_{n+1}(\mathtt{x}) - T_n(x),\ T_0(\mathtt{x}) = 1,\ T_1(\mathtt{x}) = \mathtt{x}\ .$$

(1) Écrire une procédure `tcheb` qui prend comme argument l'entier n, et qui renvoie la valeur de $T_n(\mathtt{x})$;

(2) Calculer les premiers polynômes de Tchebichev pour $n = 0, 1, 2, 3, 4, 5, 9$ et 10, vérifier la propriété $(\mathcal{P})$.

Date: 22-01-2009 18:37.

Vous utiliserez les commandes : `expand,sort, eval(...,x=cos(a))`, `combine(...,trig)`.

Exerice 3 : Factorielle.

(1) En utilisant une boucle `for`, vous écrivez une procédure `facto` qui prend comme argument l'entier n et qui renvoie la valeur $n!$ (sans utiliser les commandes $n!$ et `factorial(n)`) ;

(2) Calculer `facto(n)` pour $n = 0, \dots, 5$;

(3) Écrire une procédure `factorec` qui prend comme argument l'entier n et qui renvoie la valeur de $n!$. À la différence de la procédure `facto`, l'algorithme doit être récursif, c'est-à-dire que la procédure s'appelle elle-même comme dans la définition récurrente de $n!$ par $0! = 1$ et $(n+1)! = (n+1)n!$;

(4) Calculer `factorec(n)` pour $n = 0, \dots, 5$.

Exercice 4 : Suite de Fibonacci. La célèbre suite de Fibonacci est définie par la récurrence d'ordre 2 suivante :

$$u_{n+2} = u_{n+1} + u_n, \; u_0 = 0, \; u_1 = 1 \; .$$

Écrire une procédure qui prend un entier n et qui renvoie la valeur de u_n :

(1) de façon itérative (c'est-à-dire à l'aide d'une boucle `for` ou `while`) ;

(2) de façon récursive ;

(3) Pour chacune des deux versions vous calculerez les valeurs u_n pour $n = 0, \dots, 10$.