Suite de Fibonacci et algorithme des puissances

On considère la suite de Fibonacci, défine par

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

1. Écrire la procedure Maple

```
> fibo0 := proc(n)
>  if n <= 1 then n
>  else fibo0(n-1) + fibo0(n-2)
>  fi
> end;
```

   qui est la transcription immédiate de cette définition par récurrence. Démontrer que le cout $c(n)$ de l'appel `fibo0(n)` est proportionnel à u_n.

2. Vérifier qu'en utilisant l'option `remember`,

```
> fibo1 := proc(n) option remember;
>  if n <= 1 then n
>  else fibo1(n-1) + fibo1(n-2)
>  fi
> end;
```

   le temps de calcul devient $O(n)$. Qu'observez vous pour de grandes valeurs de n ?

3. Ecrire, en utilisant l'algorithme des puissances, la procédure

```
> expm := proc(x,n,Id,multiply)
```

   dont les arguments sont x appartenant à un ensemble muni d'une opération associative notée multiplicativement, `multiply`, dont l'élément neutre est `Id`, et `n` un entier naturel. Cette procédure rend $\mathtt{x}^{\mathtt{n}}$.

4. Soit A la matrice

$$A = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

   Montrer que, pour tout $n \geq 1$, on a

$$\begin{pmatrix} u_{n+1} \\ u_n \end{pmatrix} = A \begin{pmatrix} u_n \\ u_{n-1} \end{pmatrix}.$$

   En déduire une procédure

```
> fibo2 := proc(n)
```

   qui calcule u_n avec un nombre d'opérations proportionnel à $\log n$ (le temps de calcul est plus long car les opérations se font sur des entiers de plus en plus grands).

5. Ecrire la procédure
   > `fibo3 := proc(n,p)`

   qui calcule $u_n \pmod p$ en temps $O(\log n)$.

6. En déduire la valeur de $u_{10^{20}} \pmod{1000}$.

L'algorithme de Karatsuba pour la multiplication des polynômes

1

On s'intéresse ici à la multiplication des polynômes. Vérifier que la multiplication de deux polynômes pleins de degrés n et m coute $O(nm)$. En utilisant la formule

$$(A_1X^m + A_0)(B_1X^m + B_0) = \\ (X^{2m} + X^m)A_1B_1 - X^m(A_1 - A_0)(B_1 - B_0) + (X^m + 1)A_0B_0$$

écrire une procdure

> `karatsuba(A,B,X,m)`

qui reçoit deux polynomes en X de degrés plus petits que m, où m est une puissance de 2, et rend le produit AB.

2

Montrer que la complexité de cette procédure est en $O(n^{\frac{\log 3}{\log 2}})$. Pour tester votre procédure vous pourrez utiliser la fonction `randpoly` avec ses options (`degree = .., dense`); pour mesurer les temps vous pourrez utiliser la fonction `time` . Vérifier que le temps de calcul de la multiplication de deux polynômes pleins de degré n, et bien en $O(n^{\log 3/\log 2})$. Vérifier aussi que la multiplication de Maple est (probablement, je n'ai pas eu le temps de vérifier) plus rapide que ce que vous avez écrit ! Les opérations de base sur les polynômes sont bien optimisées dans Maple.