Université de Bordeaux — **Algèbre et calcul formel – Master**
Mathématiques — Année 2017–2018

FEUILLE D'EXERCICES n° 2

Exercice 1 – [Opérations sur les polynômes]
Soit R un anneau commutatif. Soient P et Q deux polynômes de $R[x]$, de degrés respectifs m et n. On note $P = \sum p_i x^i$ et $Q = \sum q_i x^i$, que l'on code par des listes $P = [p_0, \dots, p_{m-1}]$ et $Q = [q_0, \dots, q_{n-1}]$.

1) Écrire un algorithme `Plus(P,Q)` qui additionne P et Q. Quelle est la complexité algébrique de cet algorithme ?

2) Écrire un algorithme `Decalage(P,i)` qui multiplie P par x^i.

3) Écrire un algorithme `MultScalaire(P,a)` qui multiplie a par P (où $a \in R$). Quelle est la multiplicité algébrique de cet algorithme ?

4) Écrire un algorithme `Fois(P,Q)` qui multiplie P par Q en utilisant une multiplication similaire à celle vue dans $\mathbb{N}$. Montrer que le nombre d'opérations dans R nécessaires pour cette multiplication est inférieure ou égale à $2mn+m+n+1$.

5) Écrire un algorithme `Divise(P,Q)` (où $Q \neq 0$) qui divise P par Q en utilisant une division similaire à celle vue dans $\mathbb{N}$. Quelle est la complexité algébrique de cet algorithme ?

6) Soit q une puissance d'un nombre premier. Montrer que l'on peut effectuer la multiplication dans $\mathbb{F}_q$ avec une complexité binaire de $O((\log q)^2)$.

Exercice 2 – [Division dans $\mathbb{N}$] On rappelle l'algorithme de division des entiers donné en cours.

`Division binaire.`
`Entrée.` a, b : deux entiers naturels tels que $b \neq 0$.
`Sortie.` q, r : quotient et reste de la division euclidienne de a par b.
$a' \leftarrow a$, $s \leftarrow s(a)$, $t \leftarrow s(b)$.
Pour i de 0 à $s - t$ faire $q_i = 0$.
Tant que $a' \geqslant b$ faire :
  $b' \leftarrow 2^{s-t} b$
  Si $b' \leqslant a'$ faire :
    $q_{s-t} \leftarrow 1$
    $a' \leftarrow a' - b'$
  Sinon faire :
    $q_{s-t-1} \leftarrow 1$
    $b' = b'/2$
    $a' \leftarrow a' - b'$
  $s \leftarrow s(a')$
Sortir $q = \sum_{i=0}^{s(a)-s(b)} q_i 2^i$, $r = a'$.

Exécuter cet algorithme sur les entiers $a = 23$ et $b = 4$ puis sur $a = 23$ et $b = 7$.

Exercice 3 – [FIBONACCI]
Soit `Fib` la procédure définie récursivement par le code Sage suivant.

```
def Fib(n):
  if n<=1:
    return n
  else:
    return Fib(n-1)+Fib(n-2)
```

1) Que calcule `Fib` ?

2) Montrer que pour tout n, on a

$$\texttt{Fib}(n) = \frac{1}{\sqrt{5}}\left(\Phi^n - \Phi'^n\right),$$

où $\Phi = (1+\sqrt{5})/2$ est le nombre d'or et où $\Phi' = (1-\sqrt{5})/2$ est son conjugué.

3) Soit c_n le nombre d'additions effectuées pour calculer $\texttt{Fib}(n)$, montrer que $c_n = \texttt{Fib}(n+1) - 1$. En déduire que la complexité algébrique de `Fib` est exponentielle.

4) Proposer pour le calcul de $\texttt{Fib}(n)$ un algorithme de complexité algébrique $O(n)$ qui utilise l'égalité $\texttt{Fib}(n) = \texttt{Fib}(n-1) + \texttt{Fib}(n-2)$.

Exercice 4 – [HORNER]
Soit `Calc` la procédure définie par le code Sage suivant

```
def Calc(n,T,x):
  u = [1]
  s = 0
  for i in range(1,n+1):
    u.append(x*u[i-1])
  for i in range(0,n+1):
    s = s+u[i]*T[i]
  return s
```

où n est un entier naturel, T une liste de réels dont les indices vont de 0 à n, et où x est un réel.

1) Que calcule `Calc` ?

2) Vérifier que la procédure suivante calcule la même chose.

```
def Horn(n,T,x):
  s=T[n]
  for i in range(n-1,-1,-1):
    s=T[i]+x*s
  return s
```

3) Comparer les complexités algébriques de `Calc` et `Horn`.

Exercice 5 – [DIVISION DANS $\mathbb{N}$: PREUVE DE L'ALGORITHME]
Dans cet exercice, on donne la preuve de l'algorithme de division euclidienne binaire rappelé dans l'exercice 2.

1) Soit k le plus grand entier tel que $a \geqslant 2^k b$. Montrer que

$$k \in \{s(a) - s(b) - 1, s(a) - s(b)\}.$$

On pose $q_0 = 0$, $a'_0 = a$, $s_0 = s(a'_0) = s(a)$, $k_0 = k$ et on note a'_l, s_l et q_l les valeurs respectives de a', s et q après la l-ème exécution de la boucle "Tant que". Soit

$$q_l = \sum_{i=0}^{k} q_{l,i} 2^i$$

l'écriture binaire de q_l (il faut prendre garde que dans ces notations, ce qu'on écrit q dans l'algorithme devient q_l (où l varie) et ce qu'on écrit q_i dans l'algorithme devient $q_{l,i}$). Enfin, on note k_l le plus grand entier tel que $a'_l \geqslant 2^{k_l} b$.

On suppose qu'au rang l, $a'_l \geqslant b$, $a = bq_l + a'_l$ et $q_{l,i} = 0$ pour tout $i < k_{l-1}$.

2) Montrer que $k_l < k_{l-1}$.

3) En déduire que $q_{l+1} = q_l + 2^{k_l}$, puis que $a = bq_{l+1} + a'_{l+1}$.

4) Montrer que l'algorithme effectue la division euclidienne de a par b.