université de BORDEAUX	**ANNÉE UNIVERSITAIRE 2025–2026** **SESSION 1 D'AUTOMNE — DÉCEMBRE 2025** **MENTION :** Mathématiques et Applications **Code UE :** 4TMA701U **Intitulé de l'épreuve :** Examen de Calcul Formel **Date :** 17 Décembre 2025 **Heure :** 14 : 00 **Durée :** 3h **Documents Autorisés :** Documentation de Sage et TDs.	**Collège Sciences et Technologies** **Masters**

Cet examen comporte 3 exercices indépendants. Ils comportent tous une partie d'implémentation avec Sage. Par conséquent, vous rendrez à la fin de l'examen une copie papier ainsi qu'un fichier Sage contenant vos programmes (lisible, commenté et nettoyé si possible...) au format EX-Nom-Prenom.ipynb (feuille Jupyter) ou bien EX-Nom-Prenom.sage (fichier texte). Essayez d'être le plus lisible possible sur le choix du nom de vos fonctions, et sur les séparations entre les exercices ! Le fichier est à uploader sur Moodle.

1 Taylor Shift

L'objectif de cet exercice est de déterminer un algorithme asymptoticallement optimal pour calculer les coefficients d'un polynôme $g(X)$ défini par

$$g(X) \stackrel{\text{def}}{=} f(X+a)$$

lorsque l'on connaît $a \in \mathbb{Z}$ et les coefficients de $f \in \mathbb{Z}[X]$, de degré $n-1$.

Attention : Par défaut, et sauf mention contraire, on exprimera les complexités en nombre d'opérations **binaires**. Faites donc bien attention aux tailles des entrées et des sorties de vos algorithmes !

On notera $M(N)$ la complexité de la multiplication de deux entiers de N bits. Par exemple, si on utilise l'algorithme de Karatsuba, $M(N) = O(N^{\log_2(3)})$, et on peut descendre à $O(N \log N \log\log N)$ avec l'algorithme de Schönhage-Strassen mentionné en cours. On rappelle que les algorithmes en $O(N \log N)$ ne sont pas utilisés en pratique pour la multiplication de grands entiers.

(Q1) Montrer que si A et B sont deux polynômes de degrés n, à coefficients entiers, et dont les coefficients peuvent tous s'écrire sur ℓ bits, alors le produit $A \cdot B$ peut se calculer en $O(M(n\ell))$ opérations binaires.
Indication : *On pourra encoder A et B comme deux entiers judicieusement choisis.*

Dans un premier temps, on s'inspire de la méthode de Horner pour calculer le polynôme $g(x)$. Ainsi, on écrit g sous la forme

$$g(x) = f_0 + (x+a)\left(f_1 + (x+a)\Big(\cdots + (x+a)f_{n-1}\Big)\cdots\right)$$

Le calcul de g s'effectue alors en posant $g^{(0)} = f_{n-1}$ et à chaque étape

$$g^{(i)} = (x+a)g^{(i-1)} + f_{n-i-1}.$$

Le résultat étant alors $g^{(n-1)}$.

(Q2) Démontrer que l'algorithme spécifié ci-dessus est correct.

(Q3) Déterminer le nombre d'additions et de multiplications dans $\mathbb{Z}$ faites à chaque étape de votre algorithme.

(Q4) (i) On suppose à partir de maintenant qu'il existe une borne $B \in \mathbb{N}$ telle que tous les coefficients de f vérifient $|f_i| \leq B$. Démontrer que les coefficients g_i de $g(x)$ vérifient alors

$$|g_i| \leq B \cdot (|a| + 1)^{n-1}.$$

(ii) En déduire la taille (en bits) de la sortie g.

(Q5) On suppose que $B < 2^\ell$ pour un certain entier ℓ, et $|a| < 2^d$ pour un certain entier d. Montrer que cet algorithme réalise le calcul en $O\left(n^2 M(nd + \ell)\right)$ opérations binaires.

(Q6) Justifier que si $a = \pm 1$, cette complexité peut se réduire à seulement $O(n^2 (n + \ell))$.

Dans la suite de cet exercice, on suppose que $n = 2^m$ est une puissance de 2, et on suppose qu'on a précalculé tous les $(x + a)^{2^i}$ pour $1 \leq i < m$. On pourra par exemple supposer qu'il existe un tableau dont la case i contient la liste des coefficients de $(x + a)^{2^i}$. On se propose alors d'utiliser une approche de type « Diviser Pour Régner », de la forme suivante :

Algorithme 1 : TaylorShift

Entrées : La liste des coefficients de f et un entier a
Sorties : La liste des coefficients du polynôme $g(x) = f(x + a)$

```
1 n ← deg f + 1.
2 si n ≤ 1 alors
3 |   retourner f
4 (f_0, f_1) ← Decoupage(f)
5 g_0 ← TaylorShift(f_0, a)
6 g_1 ← TaylorShift(f_1, a)
7 g ← Recombinaison(g_0, g_1)
```

(Q7) Définir les algorithmes **Decoupage** et **Recombinaison** afin que cet algorithme soit correct, et donner une preuve de correction.

Indication : *On pourra chercher une relation judicieuse de la forme*

$$f(x) = f_0(x) + x^\ell f_1(x).$$

et exprimer g à partir de ces éléments.

(Q8) Dérouler l'algorithme à la main (sur une copie) sur l'entrée

$$f(x) = x^3 + 2x^2 + 2x + 4,$$

et l'entier $a = 3$.

On note $T(n)$ la complexité binaire de cet algorithme, et on suppose que les coefficients de f s'écrivent sur ℓ bits, et que a s'écrit sur d bits. Le but est de montrer que

$$T(n) = O\Big(\log(n) M\Big(n(nd + \ell)\Big)\Big).$$

(Q9) Écrire la relation de récurrence vérifiée par T, en fonction de $M(n)$.

(Q10) En déduire la complexité binaire de cet algorithme. Commenter sur son optimalité de cet algorithme. On pourra utiliser le fait que $2M(n/2) \leq M(n)$, ce qui est le cas pour tous les algorithmes vus dans ce cours.

Indication : *Le Théorème Maître vu en cours ne s'applique pas immédiatement ici. Le plus simple est de refaire la preuve dans ce cas. On pourra par exemple commencer par compter le nombre d'opérations dans $\mathbb{Z}$ en travaillant au niveau des polynômes, avant de convertir le tout en opérations binaires.*

(Q11) Comment peut-on effectuer le précalcul de manière efficace, tout en conservant cette complexité binaire finale ?

On se propose à présent d'implémenter cet algorithme.

(Q12) Implémenter en Sage un algorithme efficace effectuant le précalcul des $(x+a)^{2^i}$.

(Q13) Implémenter en Sage l'algorithme diviser pour régner ci-dessus.

(Q14) **Application Numérique :** On considère le polynôme

$$\Phi_{101}(x) \stackrel{\text{def}}{=} x^{100} + x^{99} + \cdots + x + 1 = \sum_{i=0}^{100} x^i.$$

Vérifiez à l'aide de votre algorithme que

$$x \cdot \Phi_{101}(x+1) = (x+1)^{101} - 1.$$

2 Algorithme de Cantor-Zassenhaus

Soit $\mathbb{F} \overset{\text{def}}{=} \mathbb{Z}/31\mathbb{Z}$ et soit $P \in \mathbb{F}[X]$ le polynôme de degré 20 dont les coefficients, rangés par degré croissant, sont :

$$(21, 30, 4, 29, 1, 15, 26, 4, 20, 2, 22, 11, 19, 0, 30, 3, 20, 0, 29, 29, 1)$$

Notre objectif va être de factoriser P.

(Q15) Calculer avec Sage les polynômes $D_i \overset{\text{def}}{=} \text{pgcd}(P, x^{31^i} - x)$ pour $1 \leq i \leq 5$. Que peut-on constater ?

(Q16) Démontrer que le quotient $\mathbb{F}[x]/(P)$ est un produit de 4 copies du corps fini $\mathbb{F}_{31^5}$.

(Q17) Rappeler pourquoi si $a \in \mathbb{F}_{31^5}^{\times}$, alors $a^{\frac{31^5-1}{2}} \in \{-1, 1\}$.

(Q18) Dans Sage, choisir au hasard un polynôme $A \in \mathbb{F}[x]$, de degré inférieur à 20, et premier avec P, puis calculer $D = \text{pgcd}(A^{\frac{31^5-1}{2}} - 1, P)$. Quelle est la probabilité que $D \in \{1, P\}$?

Indication : *Si vous avez l'impression que votre programme est très lent, c'est surement parce que vous ne calculez pas D avec la bonne complexité ! Vous pouvez expliquer pourquoi sur votre copie.*

(Q19) Continuer jusqu'à obtenir la factorisation de P.

3 Polynômes Multivariés

On cherche à résoudre dans $\mathbb{C}$ le système polynomial suivant :

$$(\mathcal{S}) \begin{cases} x^4 + x^3 + x + y^3 = 0 \\ x^3 + x^2 + xy + y^4 = 0 \\ x^2y + xy + x + y^3 = 0 \\ x^2 + xy^2 + x + y^2 = 0 \end{cases}$$

Dans cet exercise, on supposera que tous les ordres monomiaux considérés vérifient $x > y$.

(Q20) Quel est l'ordre monomial à choisir pour résoudre ce système ? Bien justifier le choix.

On se place dans l'anneau $\mathbb{Q}[x, y]$, muni de l'ordre lexicographique, avec $x > y$, et on note

$$\begin{aligned} f_0 &\stackrel{\text{def}}{=} x^4 + x^3 + x + y^3, \\ f_1 &\stackrel{\text{def}}{=} x^3 + x^2 + xy + y^4, \\ f_2 &\stackrel{\text{def}}{=} x^2y + xy + x + y^3, \\ f_3 &\stackrel{\text{def}}{=} x^2 + xy^2 + x + y^2 \end{aligned}$$

les polynômes définissant chaque équation. Soit I l'idéal $\langle f_0, f_1, f_2, f_3 \rangle$ de $\mathbb{Q}[x, y]$.

(Q21) Montrer que (f_0, f_1, f_2, f_3) n'est pas une base de Gröbner de I.

(Q22) Calculer avec Sage la base de Gröbner réduite de I.

(Q23) Déterminer toutes les solutions dans $\mathbb{C}$ du système $(\mathcal{S})$.