Université Pierre et Marie Curie — **Master M1 "Mathématiques et Applications"**
Période 2 - 2014/2015 — **4M056 – Programmation en C/C++**

TP noté de C++ - Zéros de polynômes à coefficients entiers.

- *Documents autorisés :* le polycopié de cours, les feuilles de TP et leurs corrigés, vos notes personnelles. Aucun livre extérieur n'est autorisé.
- L'usage d'Internet est **strictement** interdit pendant l'épreuve : tout accès à Internet détecté par le responsable entraînera automatiquement la note nulle.
- *Durée :* 2 heures.
- *Les fichiers sources rendus à la fin de la séance doivent contenir en commentaire les noms, prénoms et numéros des étudiants du binôme.*

Nous souhaitons résoudre des équations diophantiennes du type $P(x) = 0$ avec $x \in \mathbb{Z}$ et $P \in \mathbb{Z}[X]$.

La méthode générale de résolution est la suivante. Soit un polynome de degré d qui s'écrit $P(X) = \sum_{k=0}^{d} a_k X^k$ avec $d+1$ coefficients entiers $a_k \in \mathbb{Z}$ tels que $a_d \neq 0$ et $d \geq 1$. Il est aisé de voir que les éventuels zéros ne peuvent être que dans le segment $[-R(P), R(P)]$ avec

$$R(P) = \frac{d}{|a_d|}\left(\sup_{0\leq k\leq d-1} |a_k|\right).$$

Il suffit donc d'énumérer tous les entiers dans $[-R(P), R(P)] \cap \mathbb{Z}$ pour trouver les éventuelles solutions. Soit la classe `ZPolynome` définie par :

Classe 1

```
class ZPolynome {
        private:
                unsigned int n; // egal au degre+1
                int * coeff; //tableau dynamique des n coeffs
        public:
                ZPolynome(unsigned N,int * t);
                int eval(int) const;
                void affichage() const;
};
```

On notera que le nombre `n` de coefficients à stocker est donné par $d+1$, où d est le degré du polynôme.

Exercice 1. Le constructeur doit initialiser trivialement les deux paramètres privés à partir d'un tableau dynamique `t` de taille `N` dont on supposera que `t[N-1]` est non-nul. Écrire le code.

Y a-t-il besoin d'un constructeur par copie et d'un destructeur ? Si oui, écrire leur code ; si non, écrire en commentaire pourquoi.

Surcharger l'opérateur d'affectation qui permet d'écrire $P = Q$ pour deux polynômes.

Exercice 2. La méthode `affichage()` doit afficher un polynôme sous la forme :
`a0 + a1 X + a2 X^2 +... + a_d X^d`
sur une seule ligne dans le terminal. Écrire le code correspondant.

Pour évaluer un polynôme avec une bonne efficacité, nous introduisons l'algorithme de Hörner. Soit $x \in \mathbb{Z}$. Soit $(u_k)_{0\leq k\leq d+1}$ la suite définie par la récurrence $u_0 = 0$ et $u_{k+1} = a_{d-k} + u_k x$. Alors $u_{d+1} = P(x)$.

Exercice 3. Écrire la méthode `eval(x)` qui évalue un polynôme en x par la méthode de Hörner.

Exercice 4. Ajouter une méthode `int nb_zeros() const` qui compte le nombre de zéros entiers d'un polynôme de la class `ZPolynome` avec la méthode proposée dans l'introduction.

Exercice 5. Combien de racines entières possède le polynôme $P(X) = X^5 - 10X^4 + 19X^3 - 10X^2 + 88X + 560$?

Exercice 6. Ajouter une méthode

Racines

```
std::list<int> racines() const;
```

qui renvoie une liste chaînée (gérée par la STL) qui contient les zéros de P dans $\mathbb{Z}$. Nous rappelons que la bibliothèque `list` est nécessaire, qu'il existe, pour les listes, une méthode `push_back(x)` qui ajoute un élément à la fin d'une liste et que le constructeur par défaut d'une liste crée une liste vide.

Si vous ne souhaitez pas utiliser la STL, écrivez une méthode qui renvoie un tableau dynamique avec les zéros entiers voulus.

Attention : ne commencez pas la suite tant que ce qui précède n'a pas été fait correctement.

Exercice 7. Écrire une fonction `void affiche_liste_zeros( std::list<int> l)` qui affiche toutes les racines du polynôme. Nous rappelons pour cela l'existence d'itérateurs `std::list<int>::const_iterator It` tels que :

— `++It` fasse avancer d'un élément dans la liste,

— `*It` permette d'accéder au contenu de l'élément pointé,

— `l.begin()` et `l.end()` renvoie des pointeurs vers le début et la fin d'une liste.

Si vous ne souhaitez pas utiliser la STL, écrivez une fonction qui affiche le tableau dynamique qui contient les zéros entiers.

Exercice 8. Trouver le(s) polynôme(s) de degré 5 avec des coefficients $a_5 = 1$ et $a_k \in \{5, \ldots, 15\}$ pour $0 \leq k \leq 4$ qui possède(nt) le plus de zéros.

Nous souhaitons à présent connaître la multiplicité de chaque zéro. La multiplicité d'un zéro z d'un polynôme P de degré d est le plus petit entier k tel que $P^{(k)}(z) \neq 0$ où $P^{(k)}$ est la k-ème dérivée de P.

Exercice 9. Écrire une méthode `ZPolynome derivee() const` à la classe `ZPolynome` telle que `P.derivee()` renvoie la dérivée de P.

Exercice 10. Écrire une fonction `int multiplicite(const ZPolynome & P,int z)` qui donne la multiplicité du zéro z de P et 0 si z n'est pas un zéro de P.

Exercice 11. Quelle est la multiplicité de 3 dans $P(X) = X^7 - 12X^6 + 54X^5 - 107X^4 + 69X^3 + 54X^2 - 108X + 81$?