TP1
Polynômes

Exercice 1 Représentation creuse par coefficients

On associe à chaque polynôme la liste des couples contenant le degré et le coefficient (flottant) correspondant, ces couples étant rangés par ordre croissant selon le degré. Par exemple, le polynôme $2 - X + 5X^3$ sera codé par `[(0, 2.); (1, -1.); (3, 5.)]`. On définira donc le type polynôme par

```
type poly = (int*float) list;;
```

1 - Écrire une fonction qui calcule le coefficient de degré i d'un polynôme. Si ce polynôme ne contient pas de monôme de degré i, la fonction renverra 0.

2 - Écrire une fonction qui calcule la somme de deux polynômes.

3 - Écrire une fonction qui calcule le produit de deux polynômes avec l'algorithme de Karatsuba. On écrira d'abord les fonctions auxiliaires suivantes :

- `multCoeff : poly -> float -> poly` : l'appel `multCoeff p a` renvoie le produit du polynôme `p` par la constante `a` ;
- `degre : poly -> int` : l'appel `degre p` renvoie le degré du polynôme `p` ;
- `multXn : poly -> int -> poly` : l'appel `multXn p n` renvoie le produit du polynôme `p` par le monôme X^n ;
- `cut : poly -> int -> poly * poly` : l'appel `cut p i` renvoie le couple de polynômes (p_0, p_1) tel que $p = p_0 + X^i p_1$;
- `multNaive : poly -> poly -> poly` : l'appel `multNaive p q` renvoie le produit des polynômes `p` et `q`, calculé par l'algorithme de multiplication naïve.

4 - Écrire une fonction qui calcule le quotient et le reste de la division euclidienne de deux polynômes en utilisant la méthode de Newton. On écrira d'abord les fonctions auxiliaires suivantes :

- `renverse : int -> poly -> poly` : l'appel `renverse k p` renvoie le renversé d'ordre `k` de `p` ;
- `moduloXn : poly -> int -> poly` : l'appel `moduloXn p n` renvoie le reste de la division de `p` par le monôme X^n ;
- `inverse_mod : poly -> int -> poly` : l'appel `inverse_mod p m` renvoie l'inverse de `p` modulo X^m, obtenu par la méthode des itérations de Newton (voir cours et TD).

5 - Écrire une fonction qui évalue un polynôme en une valeur selon le schéma de Horner.

Exercice 2 Interpolation de Lagrange

Soient $n \in \mathbb{N}^*$ et $x_0, x_1, \ldots, x_n$ une suite de nombres flottants distincts. On leur associe les polynômes $L_0, L_1, \ldots, L_n$ définis pour $0 \leq j \leq n$ par

$$L_j(X) = \prod_{\substack{k=0 \\ k \neq j}}^{n} \frac{X - x_k}{x_j - x_k}.$$

Soit f une fonction définie et continue sur un intervalle I de $\mathbb{R}$. On suppose que, pour tout entier k tel que $0 \leq k \leq n$, la valeur x_k appartient à I. On montre que le le polynôme d'interpolation de Lagrange défini par

$$P_f(X) = \sum_{j=0}^{n} f(x_j) L_j(X)$$

vérifie $P_f(x_i) = f(x_i)$ pour tout $i \in \{0, 1, \ldots, n\}$.

1 - Écrire la fonction qui retourne le polynôme L_i.

2 - Écrire la fonction qui calcule le polynôme P_f.

3 - Déterminer le polynôme d'interpolation de $f : x \mapsto 64/(x^2 + 7)$ en considérant $x_0 = -5$, $x_1 = -3$, $x_2 = -1$, $x_3 = 1$, $x_4 = 3$ et $x_5 = 5$.