Comme d'habitude, le TD doit commencer par la copie des fichiers python fournis dans votre dossier personnel.

I Polynômes

1.1 Encodage

Un polynôme $P(X) = a_0 + a_1X + \cdots + a_nX^n$ est encodé en python par la liste $[a_0, \ldots, a_n]$ de longueur n + 1.

Exercice 1
Compléter les fonctions **zeros** et **elimine_zero** du fichier *polynome.py*.
Donner la raison de la présence du paramètre eps dans cette deuxième fonction.

1.2 Opérations

Exercice 2
Compléter les fonctions **somme** et **soustrait**.

On prendra garde au fait que ces opérations peuvent faire apparaître des coefficients nuls à la fin de nos polynôme. On souhaite les éliminer à chaque fois.

1.3 Evaluation

Le but est ici de calculer $P(x_0)$ de manière rapide pour un polynôme donné P et un nombre x_0.

1. Remplir la fonction **evalue_naif** qui calcule brutalement $\sum_k = 0^n a_k x_0^k$.
   Evaluer le nombre de multiplications nécessaires à ce calcul, si on note n la longueur de la liste P.
2. On se rend compte en effectuant le calcul à la main qu'on ne calcule pas tous les x_0^k en repartant de 0, mais plutôt en multipliant la puissance précédente par x_0.
   Implémenter cette idée dans la fonction **evalue**. Evaluer le nombre de multiplications

1.4 Division euclidienne

Exercice 3
Finir de compléter les fonctions présente dans *polynome.py*.

Pour rappel, quand on calcule la division euclidienne de A par B :
- on élimine dans A son dernier terme en multipliant B par aX^d où a et d sont bien choisis
- on ajoute aX^d au quotient
- on recommence avec la nouvelle valeur de A jusqu'à ce que cette opération ne soit plus possible, c'est à dire que le degré du polynôme obtenu par soustraction soit strictement inférieur à celui de B. Il s'agit alors du reste.

II Tableaux triés

Le cadre cette fois est différent : on dispose d'un tableau T de nombres que l'on souhaite trier dans l'ordre croissant.

2.1 Algorithme

On note n la longueur de T. Une première idée peut-être la suivante (tri par sélection) :

1. Trouver l'indice i_0 du plus grand élément de T et échanger T$[n-1]$ et T$[i_0]$
2. Trouver l'indice $i_0$du plus grand élément de T$[0 : n-1]$ (les n-1 derniers éléments) et échanger T$[n-1]$ et T$[i_0]$.
3. ...
4. Trouver l'indice i_0 du plus grand élément de T$[0 : 2]$ et échanger T$[i_0]$ et T$[1]$.

Un problème de cette méthode est qu'elle ne conserve pas le tableau d'origine. Nous allons donc devoir le copier.

2.1.1 En route

1. Créer un fichier **tri.py**.
2. Ecrire une fonction **copie** qui prend un tableau (ou une liste) en entrée et retourne une copie de cette liste.
3. Ecrire une fonction **echange** qui échange deux éléments d'indices donnés dans une liste.
4. Ecrire une fonction **maxi** à 3 paramètres : une liste L et deux indices j_0, j_1 et qui retourne l'indice du plus grand élément de la liste considérée entre les indices j_0 et j_1 (compris).
5. Implémenter enfin la fonction **tri** qui retourne une copie triée de la liste passée en paramètre.