PYTHON. **Méthode de Karatsuba pour la multiplication des polynômes.**

On suppose les polynômes donnés par des listes : Le polynôme $U(X) = u_0 + u_1X + ... + u_dX^d$ est donné sous la forme d'une liste $U = [u_0, u_1, ..., u_d, 0, 0, ..., 0]$ de longueur $n \geq 1 + d$, où $d = \deg U$.

1) Méthode naïve

La multiplication classique de deux polynômes représentés par des listes de longueur n nécessite $O(n^2)$ opérations.
On suppose ici de plus que les deux listes `U` et `V` sont de même longueur n, et que $\deg U + \deg V < n$.

Le produit $W = UV$ est obtenu par : $\forall k < n$, $w_k = \sum_{j=0}^{k} u_j v_{k-j}$.

Ecrire une fonction `produit(U,V)` qui étant donnés deux polynômes renvoie une liste représentant $W = UV$.

2) Méthode de Karatsuba

a) Définir la fonction `oppose(U)` qui prend comme argument un polynôme U et renvoie son opposé $-U$.

Définir la fonction `addition(U,V)` qui prend comme arguments deux polynômes U et V, et renvoie $(U + V)$. On prendra garde au fait que les listes associées aux polynômes U et V ne sont pas nécessairement de même longueur.

Définir la fonction `decalage` qui prend comme arguments un polynôme U et un entier e, et renvoie UX^e.

Donner la complexité des trois fonctions précédentes (en fonction de la longueur maximale n des listes).

b) Les notations $\lfloor x \rfloor$ et $\lceil x \rceil$ désignent respectivement la partie entière inférieure et la partie supérieure d'un nombre réel x. On peut noter que pour tout entier n, on a $\lfloor \frac{n}{2} \rfloor + \lceil \frac{n}{2} \rceil = n$.

Soient deux polynômes U et V donnés par deux listes $[u_0, u_1, ..., u_{n-1}]$ et $[v_0, v_1, ..., v_{n-1}]$ de même longueur n.

On pose $e = \lfloor \frac{n}{2} \rfloor$, puis $U_b = [u_0, u_1, ..., u_{e-1}]$ et $U_h = [u_e, u_{e+1}, ..., u_{n-1}]$ de sorte que $U = U_b + X^e U_h$.

On définit de même V_b et V_h, les parties "basse" et "haute" du polynôme V.

On a ainsi $UV = (U_bV_b) + (U_hV_b + U_bV_h)X^e + X^{2e}U_hV_h$. Expliquer comment on peut déduire UV en utilisant les fonctions définies au a) et en calculant uniquement les trois produits U_bV_b, U_hV_h et $(U_b + U_h)(V_b + V_h)$.

c) Ecrire la fonction `partition(U)` qui prend comme argument un polynôme U et renvoie le couple (U_b, U_h).

d) Ecrire la fonction `produit(U,V)` correspondante (les listes données en arguments étant de même longueur).

3) Complexité de la méthode de Karatsuba

On note $c(n)$ le nombre d'opérations effectuées pour calculer le produit de deux polynômes de degré $\leq n$.
L'objectif de cette question est de prouver que $c(n) = O(n^\alpha)$, où $\alpha = \dfrac{\log 3}{\log 2}$, qui vaut environ 1.6 (et donc < 2).
a) Justifier brièvement que $c(n) \leq c(\lfloor \frac{n}{2} \rfloor) + 2c(\lceil \frac{n}{2} \rceil) + kn$, où k est une constante qu'on n'explicitera pas.

Remarque : En particulier, si n est pair, on a $c(n) \leq 3c(\frac{n}{2}) + kn$. On considère par ailleurs que $c(1) = 1$.

b) On suppose $n = 2^p$. Majorer $c(2^p)$ en fonction de p et de k.

En notant que $3^p = n^\alpha$, en déduire que $c(n) \leq (1 + 2k)n^\alpha$.

c) On se place désormais dans le cas où n n'est pas nécessairement une puissance de 2.

On admet que $n \longmapsto c(n)$ est une fonction croissante (ce qui résulte en fait d'une récurrence forte).

On pose $p = \lfloor \log n \rfloor$. Justifier que $2^p \leq n < 2^{p+1}$. En déduire que $c(n) \leq c(2^{p+1}) \leq (1 + 2k)2^\alpha n^\alpha$.

Conclusion : On obtient ainsi $c(n) = O(n^\alpha)$.