En cas d'erreur de corrigé, n'hésitez pas à me le signaler.

Contexte

L'objectif de ce devoir est d'étudier l'ordre lexicographique. Il s'agit d'une généralisation de l'ordre d'un dictionnaire: étant donné un ordre sur un ensemble quelconque, on peut définir par récurrence un ordre sur tous les mots (vus comme produits cartésiens) constitués d'éléments de cet ensemble.

...
ÉLITE
ÉLITISME
ÉLITISTE
ÉLIXIR
ELLE
ELLÉBORE
ELLIPSE
ELLIPSOÏDAL
ELLIPSOÏDE
ELLIPTIQUE
ELLIPTIQUEMENT
ÉLOCUTION
ÉLODÉE
ÉLOGE
ÉLOGIEUSEMENT
ÉLOGIEUX
...

1. Ordre lexicographique

1.I. Ordre lexicographique sur un produit

Soient $(E, \leq_E)$ et $(F, \leq_F)$ deux ensembles ordonnés.
Pour x_1 et x_2 deux éléments de E on note $x_1 <_E x_2$ ssi $x_1 \leq_E x_2$ et $x_1 \neq x_2$.
On définit un ordre $\preccurlyeq_{E\times F}$ sur $E \times F$ par

$$\forall ((e_1, f_1), (e_2, f_2)) \in (E \times F)^2, \quad (e_1, f_1) \preccurlyeq_{E\times F} (e_2, f_2) \Leftrightarrow (e_1 <_E e_2 \text{ ou } (e_1 = e_2 \text{ et } f_1 \leq_F f_2)).$$

1.1. Montrer que $(E \times F, \preccurlyeq_{E\times F})$ est un ensemble ordonné.

La relation $\preccurlyeq_{E\times F}$ est bien une relation sur $E \times F$.
On commence par remarquer que pour tous $(e_1, f_1), (e_2, f_2) \in E \times F$ **si** $(e_1, f_1) \preccurlyeq_{E\times F} (e_2, f_2)$ **alors** en particulier $e_1 \leq_E e_2$ (car $e_1 <_E e_2$ ou $e_1 = e_2$).

- **Réflexivité**: soit $(e_1, f_1) \in E \times F$.
  Alors $e_1 = e_1$ et $f_1 \leq f_1$ d'où $(e_1, f_1) \preccurlyeq_{E\times F} (e_1, f_1)$.
- **Transitivité**: soient (e_1, f_1), (e_2, f_2) et (e_3, f_3) des éléments de $E \times F$. Supposons que $(e_1, f_1) \preccurlyeq_{E\times F} (e_2, f_2)$ et $(e_2, f_2) \preccurlyeq_{E\times F} (e_3, f_3)$.
  On procède par disjonction de cas
  ▷ Si $e_1 <_E e_2$ alors comme en particulier $e_2 \leq_E e_3$ on en déduit $e_1 <_E e_3$ et donc $(e_1, f_1) \preccurlyeq_{E\times F} (e_3, f_3)$.
  ▷ Si $e_2 <_E e_3$ alors comme en particulier $e_1 \leq_E e_2$ on en déduit $e_1 <_E e_3$ et donc $(e_1, f_1) \preccurlyeq_{E\times F} (e_3, f_3)$.
  ▷ Si $e_1 = e_2$ et $e_2 = e_3$.
  Alors $e_1 = e_3$.
  De plus $f_1 \leq_F f_2$ et $f_2 \leq_F f_3$. Donc $f_1 \leq_F f_3$. Par conséquent $(e_1, f_1) \preccurlyeq_{E\times F} (e_3, f_3)$.
  Dans tous les cas $(e_1, f_1) \preccurlyeq_{E\times F} (e_3, f_3)$.
- **Antisymétrie**: soient (e_1, f_1) et (e_2, f_2) des éléments de $E \times F$. Supposons $(e_1, f_1) \preccurlyeq_{E\times F} (e_2, f_2)$ et $(e_2, f_2) \preccurlyeq_{E\times F} (e_1, f_1)$.
  En particulier $e_1 \leq_E e_2$ et $e_2 \leq_E e_1$ d'où $e_1 = e_2$.
  Par conséquent comme $\neg(e_1 < e_2)$, on a $f_1 \leq_F f_2$.
  De même $f_2 \leq_F f_1$.
  Et donc on a bien $(e_1, f_1) = (e_2, f_2)$.

Ainsi $\boxed{(E \times F, \preccurlyeq_{E\times F}) \text{ est un ensemble ordonné}}$.

1.2. Montrer que **si** $\leq_E$ est un ordre total sur E et $\leq_F$ est un ordre total sur F **alors** $\preccurlyeq_{E\times F}$ est un ordre total sur $E \times F$.

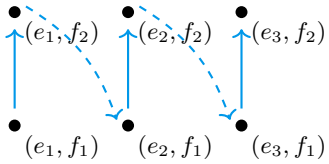
Supposons que $\leq_E$ est un ordre total sur E et $\leq_F$ est un ordre total sur F.
Soient (e_1, f_1) et (e_2, f_2) deux éléments de $E \times F$.
Comme l'ordre $\leq_E$ est **total** on a $e_1 <_E e_2$ ou $e_1 = e_2$ ou $e_2 <_E e_1$. On procède par disjonction de cas.

- Si $e_1 <_E e_2$ alors $(e_1, f_1) \preccurlyeq_{E\times F} (e_2, f_2)$.
- Si $e_2 <_E e_1$ alors $(e_2, f_2) \preccurlyeq_{E\times F} (e_1, f_1)$.
- Si $e_1 = e_2$. Comme l'ordre $\leq_F$ est total, on a $f_1 \leq_F f_2$ ou $f_2 \leq_F f_1$. Et donc $(e_1, f_1) \preccurlyeq_{E\times F} (e_2, f_2)$ ou $(e_2, f_2) \preccurlyeq_{E\times F} (e_1, f_1)$.

Dans tous les cas $(e_1, f_1) \preccurlyeq_{E\times F} (e_2, f_2)$ ou $(e_2, f_2) \preccurlyeq_{E\times F} (e_1, f_1)$.
L'ordre $\preccurlyeq_{E\times F}$ est donc $\boxed{\text{total}}$.

1.3. Faire un schéma illustrant l'ordre $\preccurlyeq_{E\times F}$.

Par exemple pour $E=\{e_1,e_2,e_3\}$ et $F=\{f_1,f_2\}$ munis des ordres totaux $e_1<_E e_2<_E e_3$ et $f_1<_F f_2$, l'ordre $\preccurlyeq_{E\times F}$ parcourt $E\times F$ «colonne par colonne»:



c'est à dire

$$(e_1,f_1)\preccurlyeq_{E\times F}(e_1,f_2)\preccurlyeq_{E\times F}(e_2,f_1)\preccurlyeq_{E\times F}(e_2,f_2)\preccurlyeq_{E\times F}(e_3,f_1)\preccurlyeq_{E\times F}(e_3,f_2).$$

1.II. Ordre lexicographique sur les mots

Soit $(E,\leq)$ un ensemble ordonné.
On définit par récurrence sur $n\in\mathbb{N}^*$ un ordre $\preccurlyeq_n$ sur E^n par

- Pour tout $(x_1,y_1)\in(E^1)^2$,
$$x_1\preccurlyeq_1 y_1 \Leftrightarrow x_1\leq y_1.$$
- Pour tout $n\geq 1$, pour tout $\left((x_k)_{k\in[\![1,n+1]\!]},(y_k)_{k\in[\![1,n+1]\!]}\right)\in(E^{n+1})^2$,
$$(x_k)_{k\in[\![1,n+1]\!]}\preccurlyeq_{n+1}(y_k)_{k\in[\![1,n+1]\!]}\Leftrightarrow\left(x_1<y_1 \text{ ou } \left(x_1=y_1 \text{ et } (x_k)_{k\in[\![2,n+1]\!]}\preccurlyeq_n(y_k)_{k\in[\![2,n+1]\!]}\right)\right).$$

1.4. Montrer que pour tout $n\in\mathbb{N}^*$, $(E^n,\preccurlyeq_n)$ est un ensemble ordonné.

On procède par **récurrence** sur $n\in\mathbb{N}^*$.
Pour $n\in\mathbb{N}^*$, notons $\mathcal{H}_n$ la proposition

$$\mathcal{H}_n: \quad (E^n,\preccurlyeq_n) \text{ est un ensemble ordonné.}$$

- **Initialisation**: Pour $n=1$ on a bien $(E^1,\preccurlyeq_1)=(E,\leq)$ est un ensemble ordonné.
- **Hérédité**: Soit $n\in\mathbb{N}^*$ et supposons $(E^n,\preccurlyeq_n)$ est un ensemble ordonné.
  Montrons que $\preccurlyeq_{n+1}$ est une relation d'ordre sur E^{n+1}.
  Première méthode:
  On vérifie les trois propriétés d'une relation d'ordre.
  - **Réflexivité** soit $(x_1,\dots,x_{n+1})\in E^{n+1}$.
    On a $x_1=x_1$ et par réflexivité de $\preccurlyeq_n$ (hypothèse de récurrence), $(x_2,\dots,x_{n+1})\preccurlyeq_n(x_2,\dots,x_{n+1})$.
    Donc $(x_1,\dots,x_{n+1})\preccurlyeq_{n+1}(x_1,\dots,x_{n+1})$.
  - **Transitivité**: soient $(x_1,\dots,x_{n+1})$, $(y_1\dots,y_{n+1})$ et $(z_1,\dots,z_{n+1})$ des éléments de E^{n+1}. Supposons que $(x_1,\dots,x_{n+1})\preccurlyeq_{n+1}(y_1\dots,y_{n+1})$ et $(y_1\dots,y_{n+1})\preccurlyeq_{n+1}(z_1,\dots,z_{n+1})$.
    On procède par disjonction de cas
    - Si $x_1<y_1$ alors comme $y_1\leq z_1$ (car $y_1<z_1$ ou $y_1=z_1$) on en déduit $x_1<z_1$ et donc $(x_1,\dots,x_{n+1})\preccurlyeq_{n+1}(z_1,\dots,z_{n+1})$.
    - De même, si $y_1<z_1$ alors comme $x_1\leq y_1$ on en déduit $x_1<z_1$ et donc $(x_1,\dots,x_{n+1})\preccurlyeq_{n+1}(z_1,\dots,z_{n+1})$.
    - Si $x_1=y_1$ et $y_1=z_1$.
      Alors $x_1=z_1$.
      De plus on a alors $(x_2,\dots,x_{n+1})\preccurlyeq_n(y_2,\dots,y_{n+1})$ et $(y_2,\dots,y_{n+1})\preccurlyeq_n(z_2,\dots,z_{n+1})$. Et alors par transitivé de $\preccurlyeq_n$ (hypothèse de récurrence) on en déduit $(x_2,\dots,x_{n+1})\preccurlyeq_n(z_2,\dots,z_{n+1})$.
      D'où $(x_1,\dots,x_{n+1})\preccurlyeq_{n+1}(z_1,\dots,z_{n+1})$.
    Dans tous les cas $(x_1,\dots,x_{n+1})\preccurlyeq_{n+1}(z_1,\dots,z_{n+1})$.
  - **Antisymétrie**:
    Soient $(x_1,\dots,x_{n+1})$ et $(y_1\dots,y_{n+1})$ des éléments de E^{n+1}.
    Supposons $(x_1,\dots,x_{n+1})\preccurlyeq_{n+1}(y_1\dots,y_{n+1})$ et $(y_1\dots,y_{n+1})\preccurlyeq_{n+1}(x_1,\dots,x_{n+1})$.
    En particulier $x_1\leq y_1$ et $y_1\leq x_1$ d'où par antisymétrie de $\leq$, $x_1=y_1$.
    On en déduit donc $\neg(x_1<y_1)$ d'où $(x_2,\dots,x_{n+1})\preccurlyeq_n(y_2,\dots,y_{n+1})$ et de même $(y_2,\dots,y_{n+1})\preccurlyeq_n(x_2,\dots,x_{n+1})$.
    Et donc par antisymétrie de $\preccurlyeq_n$ (hypothèse de récurrence), $(x_2,\dots,x_{n+1})=(y_2,\dots,y_{n+1})$.
    On a donc bien $(x_1,\dots,x_{n+1})=(y_1\dots,y_{n+1})$.
  On a donc bien $(E^{n+1},\preccurlyeq_{n+1})$ est un ensemble ordonné.
  Deuxième méthode:
  On a $\leq$ un ordre sur E et, par hypothèse de récurrence, $\preccurlyeq_n$ un ordre sur E^n.
  De plus avec les notations de la question **1.1.** on a $\preccurlyeq_{n+1}=\preccurlyeq_{E\times E^n}$ et donc par la question **1.1.**, $\preccurlyeq_{n+1}$ est un ordre sur E^{n+1}.
- **Conclusion**: on a donc montré que $\mathcal{H}_1$ et que pour tout $n\in\mathbb{N}^*$, $\mathcal{H}_n\Rightarrow\mathcal{H}_{n+1}$. Par récurrence on a donc

$$\boxed{\forall n\in\mathbb{N}^*,\ (E^n,\preccurlyeq_n) \text{ est un ensemble ordonné}}.$$

1.5. Montrer que **si** $\leq$ est un ordre total sur E, **alors** pour tout $n\in\mathbb{N}^*$, $\preccurlyeq_n$ est un ordre total sur E^n.

Supposons que l'ordre $\leq$ est total sur E. On procède par récurrence sur $n \in \mathbb{N}^*$.
- **Initialisation**: On a bien $\preccurlyeq_1 = \leq$ est un ordre total sur $E^1 = E$.
- **Hérédité**: Soit $n \in \mathbb{N}^*$ et supposons que l'ordre $\preccurlyeq_n$ est total sur E^n.
  Première méthode:
  On vérifie la définition d'un ordre total.
  Soient $(x_1, \dots, x_{n+1})$ et $(y_1, \dots, y_{n+1})$ des éléments de E^{n+1}.
  Supposons
  $$\neg\left(x_1 < y_1 \text{ ou } (x_1 = y_1 \text{ et } (x_2, \dots, x_{n+1}) \preccurlyeq_n (y_2, \dots, y_{n+1}))\right)$$
  c'est à dire
  $$(\neg(x_1 < y_1) \text{ et } (x_1 \neq y_1 \text{ ou } \neg((x_2, \dots, x_{n+1}) \preccurlyeq_n (y_2, \dots, y_{n+1})))).$$
  Comme $\leq$ est **total** on a $\neg(x_1 < y_1) \Leftrightarrow y_1 \leq x_1$ (ce n'est pas le cas pour les ordres non totaux). On procède ensuite par disjonction de cas.
  ▷ Si $y_1 < x_1$ alors $(y_1, \dots, y_{n+1}) \preccurlyeq_{n+1} (x_1, \dots, x_{n+1})$.
  ▷ Si $y_1 = x_1$, alors on en déduit $\neg((x_2, \dots, x_{n+1}) \preccurlyeq_n (y_2, \dots, y_{n+1}))$ et comme par hypothèse de récurrence $\preccurlyeq_n$ est total, on a en particulier $((y_2, \dots, y_{n+1}) \preccurlyeq_n (x_2, \dots, x_{n+1}))$ et donc $(y_1, \dots, y_{n+1}) \preccurlyeq_{n+1} (x_1, \dots, x_{n+1})$.
  Dans tous les cas on a bien $(y_1, \dots, y_{n+1}) \preccurlyeq_{n+1} (x_1, \dots, x_{n+1})$.
  On a donc montré que
  $$\neg((x_1, \dots, x_{n+1}) \preccurlyeq_{n+1} (y_1, \dots, y_{n+1})) \Rightarrow ((y_1, \dots, y_{n+1}) \preccurlyeq_{n+1} (x_1, \dots, x_{n+1}))$$
  c'est à dire que
  $$(x_1, \dots, x_{n+1}) \preccurlyeq_{n+1} (y_1, \dots, y_{n+1}) \text{ ou } (y_1, \dots, y_{n+1}) \preccurlyeq_{n+1} (x_1, \dots, x_{n+1})$$
  et ce quels que soient $(x_1, \dots, x_{n+1})$ et $(y_1, \dots, y_{n+1})$ dans E^{n+1}.
  Donc l'ordre $\preccurlyeq_{n+1}$ est total.
  Deuxième méthode:
  On a $\leq$ un ordre total sur E et, par hypothèse de récurrence, $\preccurlyeq_n$ un ordre total sur E^n.
  De plus avec les notations de la question **1.2.** on a $\preccurlyeq_{n+1} = \preccurlyeq_{E \times E^n}$ et donc par la question **1.2.**, $\preccurlyeq_{n+1}$ est total.
- **Conclusion**: la propriété est initialisée et héréditaire, donc par récurrence
  $$\boxed{\forall n \in \mathbb{N}^*,\ \preccurlyeq_n \text{ est un ordre total sur } E^n}.$$

On s'est précédemment restreint à comparer des n-uplets **de même longueur**. On peut en fait définir un ordre lexicographique sur l'ensemble des mots, c'est à dire pour des tailles finies quelconques et non nécessairement égales.
Plus précisément on pose $E^\star = \bigcup_{n \geq 1} E^n$ et on définit un ordre $\preccurlyeq_\star$ sur $E^\star$ par, pour tout $((a_1, \dots, a_p), (b_1, \dots, b_q)) \in (E^\star)^2$,

$$(a_1, \dots, a_p) \preccurlyeq_\star (b_1, \dots, b_q) \Leftrightarrow \begin{cases} (a_1, \dots, a_{\min(p,q)}) \prec_{\min(p,q)} (b_1, \dots, b_{\min(p,q)}) \\ \text{ou} \\ p \leq q \text{ et } (a_1, \dots, a_p) = (b_1, \dots, b_p) \end{cases}$$

On admet que $(E^\star, \preccurlyeq_\star)$ est bien un ensemble ordonné et que si $\leq$ est un ordre total sur E, alors $\preccurlyeq_\star$ est un ordre total sur $E^\star$.

1.6. On munit $\mathbb{N}$ de l'ordre usuel $\leq$.
Parmi les éléments suivants de $\mathbb{N}^\star$ (mots sur $\mathbb{N}$), ordonner ceux qui sont ordonnables par ordre lexicographique.

$$(1,2,3,6) \quad (1,2,4,1) \quad (1,2,1) \quad (1,3,0) \quad (2) \quad (1,2,3) \quad (1,2)$$

L'ordre $\leq$ étant un ordre total sur $\mathbb{N}$, l'ordre lexicographique $\preccurlyeq_\star$ est un ordre total sur $\mathbb{N}^\star$.
On a l'ordre suivant:

$$\boxed{(1,2) \preccurlyeq_\star (1,2,1) \preccurlyeq_\star (1,2,3) \preccurlyeq_\star (1,2,3,6) \preccurlyeq_\star (1,2,4,1) \preccurlyeq_\star (1,3,0) \preccurlyeq_\star (2)}.$$

1.7. Commenter l'appellation «ordre lexicographique».

Il s'agit bien de l'ordre usuel des dictionnaires (avec comme ordre de base l'ordre alphabétique).
En effet pour «ordonner» deux mots on compare d'abord les premières lettres. Si elles sont strictement ordonnables par l'ordre alphabétique cela permet d'ordonner les mots. Si les deux premières lettres sont identiques on compare ensuite le reste du mot selon la même méthode.

1.III. Un peu d'informatique

Les questions suivantes sont indépendantes.
On accordera une attention particulière à la présentation et l'explication du code.
On pourra utiliser au choix des fonctions itératives ou des fonctions récursives.

Aucune preuve de terminaison, de correction ou de complexité n'est demandée.

1.8. Écrire une fonction python `ordre_lexico` prenant en argument deux listes de flottants ℓ_1 et ℓ_2 et renvoyant la valeur de vérité de «$\ell_1 \preccurlyeq_\star \ell_2$» où $\preccurlyeq_\star$ est l'ordre lexicographique sur les listes de flottants. (Cf énoncé pour les exemples)

On peut par exemple procéder par récursivité, initialisant au cas où une des deux listes est vide, et pour deux listes non vides comparant les deux premiers éléments et comparant la queue des listes si ceux-ci sont égaux. (Un variant de récursion étant alors la longueur de la liste la moins longue).

```
def ordre_lexico(l1, l2):
    """Prend en argument deux listes l1 et l2 et renvoie la valeur de vérité l1 <= l2 ."""
    if l1 == []:
        return True
    elif l2 == []:
        return False
    else:
        if l1[0] < l2[0]:
            return True
        elif l1[0] > l2[0]:
            return False
        else:
            return ordre_lexico(l1[1:], l2[1:])
```

Alternativement par une boucle qui transcrit exactement la définition par récurrence.

```
def ordre_lexico(l1, l2):
    """Prend en argument deux listes l1 et l2 et renvoie la valeur de vérité l1 <= l2 ."""
    p, q = len(l1), len(l2)
    for i in range(min(p, q)):
        if l1[i] < l2[i]:
            return True
        elif l1[i] > l2[i]:
            return False
    return p <= q
```

1.9. Écrire une fonction `liste_lexico` prenant en argument un entier n et un entier p et renvoyant la liste de tous les éléments de $[\![0, n-1]\!]^p$ triés par ordre lexicographique. (Cf énoncé pour les exemples)

On peut de même procéder récursivement

```
def liste_lexico(n, p):
    """Réponse par récursivité à la question 2.2"""
    if p == 0:
        return [[]]
    L_prec = liste_lexico(n, p - 1)
    L = []
    for k in range(n):
        for q in L_prec:
            L.append([k] + q)
    return L
```

ou itérativement en constatant qu'il s'agit des écritures en base n des entiers successifs entre 0 et $n^p - 1$ (exercice: écrire un code itératif).

1.10. (**Bonus**) Écrire une fonction `permutations_lexico` prenant en argument un entier n et renvoyant la liste de tous les mots de $[\![0, n-1]\!]^n$ où toutes les coordonnées sont distinctes, triés par ordre lexicographique. (Cf énoncé pour les exemples)

De même on peut utiliser une version récursive

```
def subs(S, k):
    """Génère les permutations par ordre lexicographique de [|0,n|]\{k} à partir de celles de
    ↪ [|0,n-1|]"""
    L = []
    for p in S:
        l = []
        for x in p:
            if x < k:
                l.append(x)
            else:
                l.append(x + 1)
        L.append(l)
```

```
        return L


def permutations_lexico(n):
    """Génère les permutations par ordre lexicographique de manière récursive"""
    if n == 0:
        return [[]]
    L = []
    S = permutations_lexico(n - 1)
    for k in range(n):
        for p in subs(S, k):
            L.append([k] + p)
    return L
```

Remarque: pour chacune des questions d'informatique il y a de très nombreuses solutions possibles (par récursivité, par itérativité, en utilisant des compréhensions de liste, etc.).
L'essentiel est de réussir à adapter la méthode qui vous semble «naturelle» et de vérifier sur les cas tests.