TP n° 12 : Listes en compréhension

JOYEUX NOËL ET BONNES VACANCES !

1 Listes en compréhension

En PYTHON, la construction de listes en compréhension (ou d'ensembles ou de dictionnaires) permet de créer des listes de manière à la fois concise et élégante. La syntaxe est très proche de celle que l'on utilise en mathématiques pour décrire un ensemble. Par exemple $\{f(x) \mid x \in A, P(x)\}$ peut s'écrire

PYTHON
```
{f(x) for x in A if P(x)}
```

1.1 Syntaxe

Pour créer la liste des i^2 pour i allant de 0 à 10, on peut écrire :

PYTHON
```
[i ** 2 for i in range(11)]
```

1. Créer la liste des 2^i pour i allant de 2 à 15.

Pour créer la liste des entiers divisibles par 3 entre 0 et 18 sauf 9, on peut écrire :

PYTHON
```
[i for i in range(19) if i % 3 == 0 and i != 9]
```

2. Proposer une autre manière de créer la liste précédente en utilisant un `range` avec trois arguments.
3. Créer la liste des entiers impairs divisibles par 7 pour i allant de 2 à 15.

La syntaxe générale est :

✔

PYTHON
```
[expr for element in iterable if conditions]
```

1.2 Filtrage

On peut utiliser cette construction pour copier une liste :

PYTHON
```
def copie(liste):
    return [element for element in liste]
```

Cela n'est pas bien utile puisqu'il suffit d'écrire `liste[:]` ou `list(liste)` qui sont plus concis. L'intérêt vient lorsque l'on souhaite faire un filtrage. Par exemple, la fonction suivant permet de renvoyer la liste des éléments positifs d'une liste (dans le même ordre relatif). C'est ce que l'on appelle le *filtrage*.

PYTHON
```
def filtre_positifs(liste):
    return [element for element in liste if element >= 0]
```

4. Écrire une fonction d'une ligne qui renvoie la liste des éléments impairs d'une liste d'entier.

5. Écrire une fonction `intersection(liste1, liste2)` qui renvoie une liste contenant les éléments qui sont à la fois dans `liste1` et `liste2` dans le même ordre relatif que dans `liste1`.

1.3 Compréhension et `for` imbriqués

On peut utiliser plusieurs `for` :

PYTHON
```
couples = [(i, j) for i in range(5) for j in range(3)]
```

Ce que l'on écrivait auparavant :

PYTHON
```
couples = []
for i in range(5):
    for j in range(3):
        couples.append((i, j))
```

6. Écrire une fonction d'une ligne qui renvoie, pour deux arguments entiers naturels n et m, la liste des couples d'entiers de $[\![0, n]\!]^2$ dont la somme est multiple de m.
7. Écrire une fonction d'une ligne qui renvoie, pour deux arguments entiers naturels n et m, le maximum du produit des couples d'entiers de $[\![0, n]\!]^2$ dont la somme est multiple de m.

1.4 Initialisation de matrices

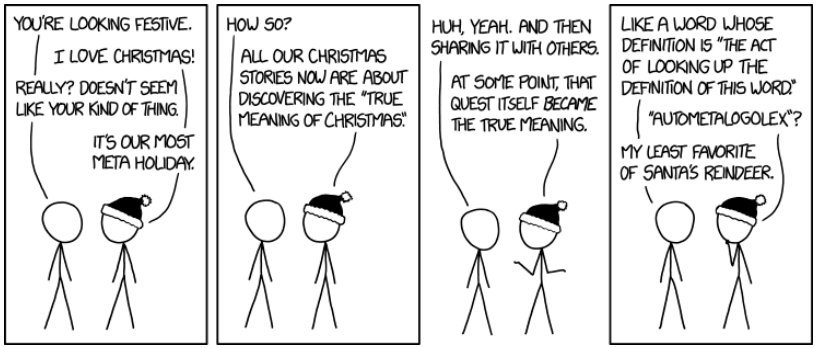
Une application très intéressant de cette construction est d'initialiser une matrice — un tableau de nombres — sous la forme d'une liste de listes.

8. Que pensez-vous de `[[0] * n] * n]` pour créer une matrice de taille $n \times n$ initialisée avec des 0 ? Vérifier sur des exemples, en modifiant une ou plusieurs cases de la matrice.
9. Que pensez-vous de `[[0 for _ in range(n)] for _ in range(n)]` ? Vérifier sur des exemples, en modifiant une ou plusieurs cases de la matrice.
10. Écrire une fonction qui renvoie une matrice de taille $n \times m$, avec n et m deux entiers naturels passés en arguments, avec, pour $0 \leq i < n$ et $0 \leq j < m$, la case (i, j) initialisée à $i + j$.
11. Modifier cette fonction pour prendre en troisième argument une fonction f au moins définie sur $\mathbb{N}^2$ telle que la case (i, j) soit initialisée à $f(i, j)$.
12. Réécrire cette fonction sans utiliser de liste en compréhension (à l'ancienne).

2 Pour aller plus loin

Plutôt que d'aborder cette partie, vous pouvez terminer le TP précédent.

13. Écrire un programme qui compte le nombre de mots dans un fichier dont le nom est passé en argument. On pourra bien sûr essayer comme nom de fichier le nom du script PYTHON lui-même, pour connaître le nombre de mots que l'on a écrit depuis le début du TP ! Quelle est sa complexité ?
14. Écrire un programme qui compte le nombre de mots *différents* dans un fichier dont le nom est passé en argument. On pourra utiliser la structure d'ensemble (`set`). Quelle est sa complexité ?
15. Écrire un programme qui renvoie une liste formée des mots et de leur nombre d'occurrences dans un fichier dont le nom est passé en argument. On pourra utiliser la structure de dictionnaire (`dict`). Quelle est sa complexité ?
16. Écrire une fonction qui renvoie le mot le plus utilisé dans un fichier dont le nom est passé en argument.
17. Écrire une fonction qui renvoie les n mots les plus utilisés dans un fichier dont le nom ainsi que n son passés en argument.
18. Lire la documentation de la bibliothèque `collections` et reprendre les trois dernières questions en utilisant la structure `collections.Counter`.



https://xkcd.com/1932/