NSI · PREMIÈRE / COURS

Tableaux et p-uplets

Astrolab Cours · astrolabcours.fr

01 Le problème

Stocker les notes de 30 élèves dans 30 variables `n1`, `n2`… est ingérable. Et une fonction qui doit renvoyer à la fois un minimum et un maximum ne peut renvoyer qu'une valeur. Il faut des valeurs qui en regroupent plusieurs.

01 L'idée : les casiers numérotés

Un tableau, c'est une rangée de casiers numérotés à partir de 0. On peut ouvrir le casier 3, changer son contenu, ajouter un casier au bout. Un p-uplet, c'est une enveloppe scellée qui contient plusieurs valeurs : on peut lire ce qu'il y a dedans, pas le changer.

01 Les p-uplets (tuples)

Un **p-uplet** est une suite **non modifiable** de valeurs, éventuellement de types différents : `point = (3, 5)`, `eleve = ("Léa", 16, "1re")`.

On lit `point[0]`, mais `point[0] = 4` provoque une erreur. Ils servent à **renvoyer plusieurs valeurs** :

```
def min_max(t):
    return min(t), max(t)

a, b = min_max([4, 9, 1])   # a = 1, b = 9
```

Les tableaux (listes Python)

Un **tableau** est une suite **modifiable** de valeurs, accessibles par leur **indice**, de `0` à `len(t) - 1`.

```
t = [12, 15, 9]
t[1] = 16          # modification
t.append(14)       # ajout à la fin
for x in t:        # parcours des valeurs
    print(x)
for i in range(len(t)):   # parcours des indices
    print(i, t[i])
```

Une **liste en compréhension** construit un tableau en une ligne :

© Astrolab Cours · astrolabcours.fr/nsi/premiere/tableaux-et-tuples/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits

```
carres = [x ** 2 for x in range(5)]      # [0, 1, 4, 9, 16]
pairs = [x for x in t if x % 2 == 0]
```

Les tableaux de tableaux

Une grille (morpion, image, matrice) est un tableau de lignes :

```
grille = [[0, 0, 0], [0, 1, 0], [0, 0, 0]]
grille[1][1]   # ligne 1, colonne 1 : vaut 1
```

Alias et copie

`b = a` ne crée pas de nouveau tableau : `a` et `b` sont deux noms pour le même. Pour copier : `b = a.copy()`. Et pour une grille, `[[0] * 3] * 3` crée trois fois la **même** ligne : on écrit `[[0] * 3 for _ in range(3)]`.

Exemple guidé

Écrire une fonction qui renvoie à la fois la plus petite et la plus grande note d'un tableau non vide.

```
def min_max(notes):
    mini = notes[0]
    maxi = notes[0]
    for n in notes:
        if n < mini:
            mini = n
        if n > maxi:
            maxi = n
    return (mini, maxi)

bas, haut = min_max([14, 9, 17, 11])
# bas vaut 9, haut vaut 17
```

1. **Le tableau** `notes` regroupe autant de valeurs qu'on veut, accessibles par leur indice.
2. On part de la première note, puis on parcourt le tableau en mettant à jour le minimum et le maximum.
3. **Le tuple** `(mini, maxi)` permet de renvoyer deux valeurs. On les récupère d'un coup avec `bas, haut = ...`.

Retour au problème

Les 30 notes tiennent dans un seul tableau, et le tuple renvoie le minimum et le maximum ensemble. Ajouter un 31e élève ne change pas une ligne de la fonction.

Les pièges

- Le dernier indice est `len(t) - 1`, pas `len(t)`.
- `b = a` est un alias, pas une copie.
- `[[0] * 3] * 3` partage la même ligne trois fois.

© Astrolab Cours · astrolabcours.fr/nsi/premiere/tableaux-et-tuples/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits

© Astrolab Cours · astrolabcours.fr/nsi/premiere/tableaux-et-tuples/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits