Exercices types construits
Correction

Christophe Viroulaud

Première - NSI

DonRep 09

Sommaire

Exercice 1

Éléments de correction :

- Un tuple n'est pas mutable (modifiable).
- Dans un dictionnaire, tenter de lire la valeur associée à une clé qui n'existe pas renvoie une erreur.
- Dans un dictionnaire, il est possible de créer un nouveau couple clé/valeur.

Sommaire

Exercice 2

```
1 tab = [0, 1, 2, 3, 4, 5]
2
3 for i in range(len(tab)):
4     tab[i] = tab[i]**2
```

Sommaire

Exercice 3

```
1  def mini(t: tuple) -> int:
2      """
3      renvoie l'entier minimum de t
4      """
5      minimum = 101
6      for elt in t:
7          if elt < minimum:
8              minimum = elt
9      return minimum
```

```
1  >>> tup = (17, 32, 8, 4, 35, 13)
2  >>> mini(tup)
3  4
```

Code 1 – Appel de la fonction

Sommaire

Exercice 4

```
1  def mediane(t: tuple) -> float:
2      """
3      calcule la médiane du tuple
4      indications:
5          le tuple est ordonné
6          le tuple contient au moins 2 éléments
7      """
8      taille = len(t)
9      milieu = taille//2
10     if taille % 2 == 0:  # pair
11         # la numérotation commence à 0
12         med = (t[milieu-1]+t[milieu])/2
13     else:  # impair
14         med = t[milieu]
15     return med
```

Observations

- Le programme respecte l'algorithme donné.
- Dans le cas pair, que se passerait-il si la taille du tuple était inférieure à 2 ?

```
1 salaire = (800, 830, 830, 950, 1002, 1100,
2     1100, 1103, 1340, 1530, 1600)
3 salaire1 = (900, 950, 950, 960, 1050, 1060,
4         1100, 1160, 1370, 1555)
5 print(mediane(salaire))
6 print(mediane(salaire1))
```

Code 2 – Appel de la fonction

Sommaire

Exercice 5

```
1  def ecart_max(t: list) -> int:
2      maxi = 0
3      # attention à ne par sortir du tableau
4      for i in range(len(t) - 1):
5          ecart = t[i+1]-t[i]
6          if ecart > maxi:
7              maxi = ecart
8      return maxi
```

Observation

Il faut remarquer le générateur **`range(len(t) - 1)`**. La boucle s'arrête à l'avant-dernier élément du tableau. Ceci permet de ne pas dépasser la taille du tableau dans l'appel **`t[i+1]`**.

Sommaire

Exercice 6

```python
def somme(t1: list, t2: list) -> list:
    tab = [0, 0, 0, 0, 0]
    for i in range(5):
        tab[i] = t1[i]+t2[i]
    return tab
```

```python
t1 = [12, 17, 8, 10, 13]
t2 = [4, 18, 9, 11, 23]
print(somme(t1, t2))
```

Code 3 – Appel de la fonction

Sommaire

Exercice 7

```
1  bibliotheque = [
2      {"titre": "Il était deux fois",
3          "auteur": "Franck Thilliez",
4          "editeur": "Poche",
5          "prix": 8.70},
6      {"titre": "Fahrenheit 451",
7          "auteur": "Ray Bradbury",
8          "editeur": "Folio",
9          "prix": 6.30}
10 ]
```

```
1  for livre in bibliotheque:
2      # livre contient un dictionnaire
3      print(livre["auteur"])
```

Code 4 – Le tableau **bibliotheque** contient des dictionnaires.

Sommaire

Exercice 8

```
groupe = [
    {"prenom": "Alice",
        "nom": "Durant",
        "notes": [12, 8, 10, 9.5]},
    {"prenom": "Bob",
        "nom": "Bois",
        "notes": [15, 17, 18, 14]},
    {"prenom": "John",
        "nom": "Doe",
        "notes": [10.5, 8, 16, 13.5]},
    {"prenom": "Jules",
        "nom": "Dupont",
        "notes": [12, 9, 17.5, 10]},
    {"prenom": "Alan",
        "nom": "Turing",
        "notes": [14, 18, 16, 19]},
]
```

```python
def moyenne(eleve: dict) -> float:
    """
    calcule la moyenne de l'élève
    """
    somme = 0
    for n in eleve["notes"]:
        somme += n

    nb_notes = len(eleve["notes"])
    return somme/nb_notes
```

```
1 # moyenne générale
2 somme_moy = 0
3 for eleve in groupe:
4     # calcule la moyenne de chaque élève
5     somme_moy += moyenne(eleve)
6
7 moyenne_generale = somme_moy/len(groupe)
8 print(moyenne_generale)
```

Code 6 – Calcul de la moyenne générale