

Exercices parcours séquentiel

Correction

Christophe Viroulaud

Première - NSI

Algo 02

```
1 les_notes = [randint(0, 20) for _ in range  
    (30)]
```

Code 1 – Construction par compréhension

1. Exercice 1

2. Exercice 2

3. Exercice 3

4. Exercice 4

5. Exercice 5

6. Exercice 6

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

Exercice 1

```
1 def extrema(tab: list) -> tuple:
2     """
3     Renvoie le mini et le maxi de tab
4
5     Args:
6         tab (list):
7     Returns:
8         tuple: (mini, maxi)
9     """
10    mini = 20
11    maxi = 0
12    for note in tab:
13        if note < mini:
14            mini = note
15        if note > maxi:
16            maxi = note
17    return (mini, maxi)
```

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

1. Exercice 1

2. Exercice 2

3. Exercice 3

4. Exercice 4

5. Exercice 5

6. Exercice 6

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

Exercice 2

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

```
1 def maxi_position(tab: list) -> tuple:
2     """
3     renvoie le max et sa première position
4     dans le tableau
5     """
6     # On peut affecter plusieurs variables
7     sur 1 ligne
8     indice, note_max = 0, 0
9     for i in range(len(tab)):
10         if tab[i] > note_max:
11             note_max = tab[i]
12             indice = i
13     return (indice, note_max)
```

1. Exercice 1

2. Exercice 2

3. Exercice 3

4. Exercice 4

5. Exercice 5

6. Exercice 6

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

Exercice 3

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

```
1 def maxi_position_dernier(tab: list) -> tuple:
2     """
3     renvoie le max et sa dernière position dans
4     le tableau
5     """
6     # On peut affecter plusieurs variables sur
7     1 ligne
8     indice, note_max = 0, 0
9     for i in range(len(tab)):
10         # il suffit juste de modifier la
11         comparaison
12         if tab[i] >= note_max:
13             note_max = tab[i]
14             indice = i
15     return (indice, note_max)
```


Sommaire

1. Exercice 1
2. Exercice 2
3. Exercice 3
4. Exercice 4
5. Exercice 5
6. Exercice 6

Exercices parcours
séquentiel
Correction

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

Exercice 4

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

```
1 def maxi_nb(tab: list) -> int:
2     """
3     renvoie le nombre d'occurences du maximum
4     """
5     nb, note_max = 0, 0
6     for note in tab:
7         if note > note_max:
8             # réinitialisation du max
9             note_max = note
10            nb = 1
11        elif note == note_max:
12            nb += 1
13    return nb
```

Sommaire

1. Exercice 1
2. Exercice 2
3. Exercice 3
4. Exercice 4
5. Exercice 5
6. Exercice 6

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

Exercice 5

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

```
1 def est_present(tab: list, note: int)->bool:
2     """
3     vérifie si note est dans le tableau
4     """
5     for n in tab:
6         if n == note:
7             return True
8     # On est sorti de la boucle sans avoir
    trouvé note
9     return False
```

La fonction dépend de la taille du tableau. La complexité est **linéaire**.

Sommaire

1. Exercice 1

2. Exercice 2

3. Exercice 3

4. Exercice 4

5. Exercice 5

6. Exercice 6

Exercices parcours
séquentiel
Correction

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6

```
1 def compter_voyelles(mot: str)->dict:
2     """
3     compte le nombre de chaque voyelles de mot
4     """
5     voyelles = {"a": 0, "e": 0, "i": 0, "o": 0, "u": 0, "y": 0}
6     list_voyelles = ["a", "e", "i", "o", "u", "y"]
7     for lettre in mot:
8         # compare la lettre aux voyelles
9         if lettre in list_voyelles:
10             voyelles[lettre] += 1
11     return voyelles
```

Code 2 – version 1

```
1 def compter_voyelles(mot: str)->dict:
2     """
3     compte le nombre de chaque voyelles de mot
4     """
5     voyelles = {"a": 0, "e": 0, "i": 0, "o": 0, "u": 0, "y": 0}
6     for lettre in mot:
7         # compare la lettre aux voyelles
8         if lettre in voyelles.keys():
9             voyelles[lettre] += 1
10    return voyelles
```

Code 3 – version 2

```
1 def max_voyelles(voyelles: dict) -> list:
2     """
3     parcourt le dict voyelles et renvoie
4     celle qui a la plus grande valeur
5     """
6     maxi = 0
7     lettres_maxi = []
8     for lettre, nb in voyelles.items():
9         if nb > maxi:
10             maxi = nb
11             # on réinitialise le tableau avec la
nouvelle lettre max
12             lettres_maxi = [lettre]
13         elif nb == maxi:
14             lettres_maxi.append(lettre)
15     return lettres_maxi
```

Exercice 1

Exercice 2

Exercice 3

Exercice 4

Exercice 5

Exercice 6