

T6.4 Spécification

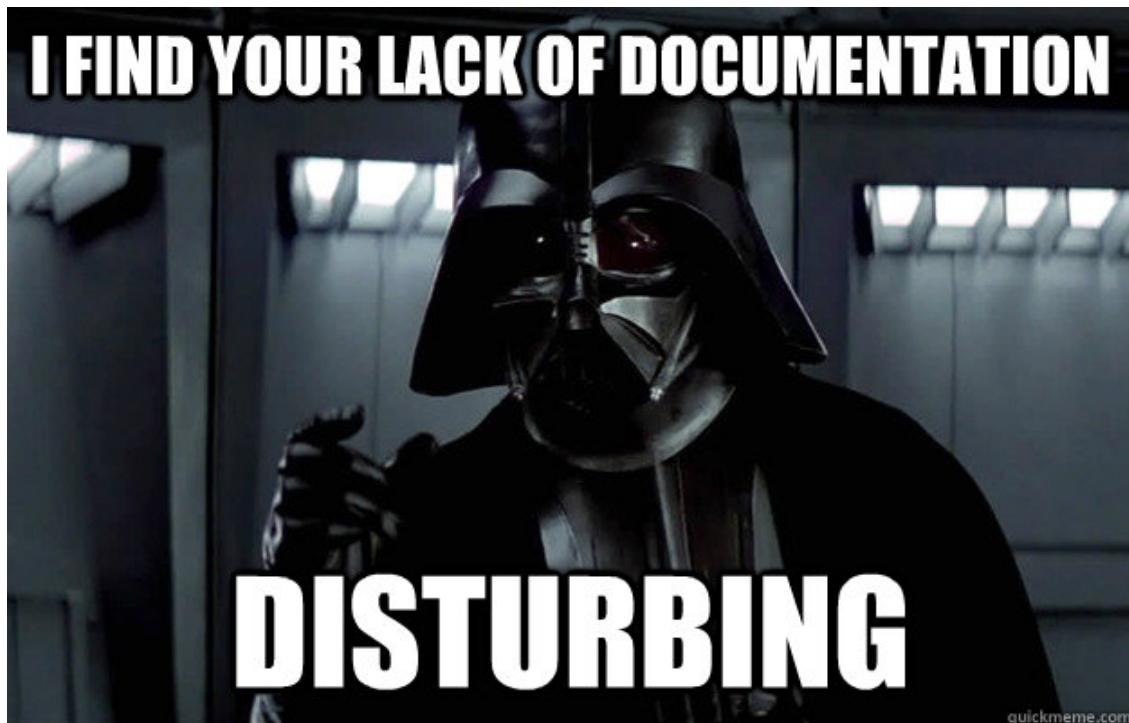
Commençons par une devinette: que fait la fonction suivante:

Script Python

```

1 def doublon(l, c):
2     n = 0
3     for e in c:
4         if e == l:
5             n += 1
6     return n > 1

```



Mieux?

Script Python

```

1 def verifie_doublon(l: str, c: str) -> bool:
2     """
3         Vérifie si la chaîne c contient au moins deux occurrences
4         de la chaîne l
5     """
6     nb_occurrences = 0
7     for element in c:
8         if element == l:
9             nb_occurrences += 1
10    return nb_occurrences > 1

```

1. 6.4.1 Spécifier et documenter

Spécification

De manière générale la **spécification** est un ensemble de d'exigences à satisfaire par un produit ou un service.

En programmation, **spécifier** un programme/une fonction revient à décrire explicitement ce qu'il/elle doit faire et dans quelles conditions.

Pour cela:

- on décrit ce que fait la fonction dans une documentation appelée **docstring**, placée au début du corps de la fonction, écrite sur plusieurs lignes entre triple quotes;
- on utilise des noms de variables explicites (voir T6.1.1);
- on précise le type de(s) paramètre(s), et éventuellement les conditions d'utilisations de la fonction: on parle de **préconditions**;
- on précise le type de la valeur renvoyée par la fonction: on parle de **postconditions**;
- ces types apparaissent dans la docstring ou en *annotations de types* (depuis Python 3.5) comme dans l'exemple ci-dessus;
- on intègre éventuellement des exemples d'utilisation qui peuvent servir de tests.

La docstring est alors disponible en utilisant la fonction `help`:

Script Python

```

1  >>> help(verifie_doublon)
2  Help on function verifier_doublon in module __main__:
3
4  verifier_doublon(l: str, c: str) -> bool
5      Vérifie si la chaîne c contient au moins deux occurrences
6      de la chaîne l
7
8  >>>

```

Intérêt

- programmer : documenter **avant** d'écrire le code donne un objectif clair;
- relire : les programmes complexes sont difficiles à comprendre. La documentation simplifie cette étape;
- collaborer : travailler à plusieurs demande de l'organisation et une documentation claire est indispensable.

PEP8 & Zen

Pour une meilleure lecture du code et une meilleure communication entre les différents collaborateurs d'un projet, il est donc nécessaire de prendre de bonnes habitudes de programmation et de respecter des conventions d'écriture:

- La [PEP8](#) (Python Enhancement Proposal 8) regroupe les bons usages
- Le Zen Python :

Script Python

```
>>> import this
```

2. 6.4.2 Exemples

Exemple sans tests

La fonction `maximum` vue aux chapitres précédentes:

Script Python

```
1 def maximum(n1, n2):
2     if n1 > n2:
3         return n1
4     else:
5         return n2
```

devient:

Script Python

```
1 def maximum(n1: int, n2: int) -> int:
2     """
3         Renvoie l'entier maximum entre n1 et n2
4         n1 et n2 : deux entiers
5     """
6     if n1 > n2:
7         return n1
8     else:
9         return n2
```

👍 Exemple avec tests

Voici la fonction `bissextile` spécifiée et documentée. On peut y ajouter des exemples/tests.

🐍 Script Python

```

1 def is_leap(year: int) -> bool:
2     """
3         Renvoie True si l'entier year correspond à une année bissextile
4         et False sinon.
5         >>> is_leap(2021)
6             False
7         >>> is_leap(2020)
8             True
9         >>> is_leap(2000)
10        True
11        >>> is_leap(1900)
12        False
13        """
14        if year%4 == 0 and year%100 != 0:
15            return True
16        elif year%400 == 0:
17            return True
18        else:
19            return False

```

On peut ensuite lancer les tests à l'aide du module `doctest`:

🐍 Script Python

```

>>> import doctest
>>> doctest.testmod()
TestResults(failed=0, attempted=4)
>>>

```

Testez cette fonction comme ci-dessus, puis en passant en argument `verbose=True` à la fonction `testmod`.

3. 6.4.3 Exercices

👍 Exercice 1

Énoncé

Documenter les fonctions `compte_voyelles` et `fizzbuzz` des chapitres précédents.

Correction