

Commentaires

Chaque fonction est accompagnée d'un commentaire appelé **docstring**. Sa syntaxe varie selon les programmeurs mais une bonne pratique consiste à :

- donner une brève description,
- préciser le type et le rôle des paramètres,
- préciser ce que renvoie la fonction.

Cette *docstring* est accessible depuis la console en écrivant **help(nom_fonction)**.

De plus une autre bonne pratique consiste à préciser le *type* des arguments de la fonction. La syntaxe

```
1 def est_pair(x: int)->bool:
```

précise que le paramètre *x* est un entier et que la fonction renvoie un *booléen*.

Exercice 1 :

```
1 def est_pair(x: int)->bool:
2     """
3     vérifie la parité
4
5     Parameters
6     -----
7     x : int
8
9     Returns
10    -----
11    boolean
12    """
13    return x%2 == 0
```

Exercice 2 :

```
1 def valeur_absolue(x: int)->int:
2     """
3     renvoie la valeur absolue de x
4     """
5     if x < 0:
6         return -x
7     else:
8         return x
```

Exercice 3 :

```
1 def surface(r: int)->float:
2     """
3     renvoie la surface du disque de rayon r
4     """
5     return 3.14*r**2
```

Exercice 4 :

```
1 def est_majeur(age: int)->bool:
2     """
3     vérifie si la personne est majeur
4
5     Parameters
6     -----
7     age : int
8         âge de la personne.
9     Returns
10    -----
11    boolean
12
13    """
14    return age >= 18
```

Exercice 5 :

```
1 def puissance(x: int, n: int)->int:
2     """
3     élève x à la puissance n
4
5     Parameters
6     -----
7     x : int
8         entier.
9     n : int
10        exposant.
11
12    Returns
13    -----
14    res : int
15    """
16    res = 1
17    for i in range(n):
18        res *= x
19    return res
```

Exercice 6 :

```
1 def pythagore(a: int, b: int, c: int)->bool:
2     """
3     vérifie si le triangle a, b, c est rectangle
4
5     Parameters
6     -----
7     a, b, c: int
8         mesures des côtés.
9
10    Returns
11    -----
12    Boolean.
```

```
13
14     """
15     return a**2 + b**2 == c**2
```

Exercice 7 :

```
1 def bissextile(annee: int)->bool:
2     """
3     vérifie si l'année est bissextile
4     """
5     # le and est évalué avant le or
6     return annee%4 == 0 and not annee%100 == 0 or annee%400 == 0
7
8 def nb_jours(annee: int)->int:
9     """
10    renvoie le nombre de jours dans annee
11    """
12    if bissextile(annee):
13        return 366
14    else:
15        return 365
16
17 def nb_jours_mois(annee: int, mois: int)->int:
18     """
19    renvoie le nombre de jours dans mois
20    (en fonction de l'année)
21    """
22    if mois == 2: #février
23        if bissextile(annee):
24            return 29
25        else:
26            return 28
27    # astuce avec la prise en compte de juillet et août
28    elif mois <= 7:
29        return 30 + mois%2
30    else:
31        return 31 - mois%2
```

Exercice 8 :

```
1 def nombres_paires(x: int)->list:
2     """
3     renvoie la liste des nombres pairs < x
4     """
5     return [i for i in range(0,x,2)]
```

Exercice 9 :

```
1 def diviseur(a: int)->list:
2     """
3     renvoie la liste des diviseurs de a
4     """
```

```
5     res = []
6     diviseur = 1
7     while diviseur < a:
8         # si le reste est nul c'est un diviseur
9         if a%diviseur == 0:
10             res.append(diviseur)
11         # passe au diviseur suivant
12         diviseur += 1
13     return res
```

Exercice 10 :

```
1 def est_premier(x: int)->bool:
2     """
3     renvoie True si x est un nombre premier
4     """
5     diviseur = 2
6     # si le reste est nul, c'est que nous avons un diviseur
7     while diviseur < x and not(x%diviseur == 0):
8         diviseur += 1
9     # On a divisé par tous les nombres < x
10    if diviseur == x:
11        return True
12    else: # on s'est arrêté avant
13        return False
```

Exercice 11 :

```
1 from random import randint
2
3 def aleatoire_100(n: int)->list:
4     """
5     renvoie une liste de n éléments compris entre 0 et 100
6     """
7     return [randint(0, 100) for _ in range(n)]
8
9 def position(tableau: list, element: int)->int:
10    """
11    renvoie l'indice de la première position de element
12
13    Parameters
14    -----
15    tableau : list
16        un tableau d'entiers
17    element : int
18        un entier.
19
20    Returns
21    -----
22    l'indice de element.
23
24    """
```

```
25     for i in range(len(tableau)):
26         if tableau[i] == element:
27             return i
28     return -1
29
30 print(position(aleatoire_100(50),10))
```

Exercice 12 :

```
1 def nb_voyelles(mot: str)->int:
2     """
3     renvoie le nombre de voyelles dans mot
4     """
5     voyelles = ["a","e","i","o","u","y"]
6     nb = 0
7     for c in mot:
8         if c in voyelles:
9             nb += 1
10    return nb
```

Exercice 13 :

```
1 import turtle as t
2
3 def triangle(c: int)->None:
4     t.begin_fill()
5     for _ in range(3):
6         t.forward(c)
7         t.left(120)
8     t.end_fill()
9
10 t.up()
11 for _ in range(3):
12     t.left(90)
13     t.forward(50)
14     t.right(90)
15     triangle(100)
16 t.done()
```