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 :

Christophe Viroulaud

```python
def est_majeur(age: int)->bool:
    """
    vérifie si la personne est majeur

    Parameters
    ----------
    age : int
        âge de la personne.
    Returns
    -------
    boolean

    """
    return age >= 18
```

Exercice 5 :

```python
def puissance(x: int, n: int)->int:
    """
    élève x àla puissance n

    Parameters
    ----------
    x : int
        entier.
    n : int
        exposant.

    Returns
    -------
    res : int
    """
    res = 1
    for i in range(n):
        res *= x
    return res
```

Exercice 6 :

```python
def pythagore(a: int, b: int, c: int)->bool:
    """
    vérifie si le triangle a, b, c est rectangle

    Parameters
    ----------
    a, b, c: int
        mesures des côtés.

    Returns
    -------
    Boolean.
```


Christophe Viroulaud

```
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_pairs(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      """
```


Christophe Viroulaud

```
    res = []
    diviseur = 1
    while diviseur < a:
        # si le reste est nul c'est un diviseur
        if a%diviseur == 0:
            res.append(diviseur)
        # passe au diviseur suivant
        diviseur += 1
    return res
```

Exercice 10 :

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

Exercice 11 :

```
from random import randint

def aleatoire_100(n: int)->list:
    """
    renvoie une liste de n éléments compris entre 0 et 100
    """
    return [randint(0, 100) for _ in range(n)]

def position(tableau: list, element: int)->int:
    """
    renvoie l'indice de la première position de element

    Parameters
    ----------
    tableau : list
        un tableau d'entiers
    element : int
        un entier.

    Returns
    -------
    l'indice de element.

    """
```


Christophe Viroulaud

```
    for i in range(len(tableau)):
        if tableau[i] == element:
            return i
    return -1

print(position(aleatoire_100(50),10))
```

Exercice 12 :

```
def nb_voyelles(mot: str)->int:
    """
    renvoie le nombre de voyelles dans mot
    """
    voyelles = ["a","e","i","o","u","y"]
    nb_voyelles = 0
    for c in mot:
        if c in voyelles:
            nb_voyelles += 1
    return nb_voyelles
```

Exercice 13 :

```
import turtle as t

def triangle(c: int)->None:
    t.begin_fill()
    for _ in range(3):
        t.forward(c)
        t.left(120)
    t.end_fill()

t.up()
for _ in range(3):
    t.left(90)
    t.forward(50)
    t.right(90)
    triangle(100)
t.done()
```


Christophe Viroulaud