

```
# -*- coding: utf-8 -*-
"""
```

```
Created on Sat Nov 16 18:56:03 2019
```

```
@author: dconduche
"""
```

```
"""trois exemples de suites récurrentes :
```

```
Exemple 1 :
```

```
u_0 = 5
```

```
u_{n+1} = 1/(n+1) * exp(-u_n)
```

```
Exemple 2 :
```

```
u_0 = 1
```

```
u_1 = 2
```

```
u_{n+2} = sqrt(u_{n+1} + 2*u_n)
```

```
Exemple 3 :
```

```
u_0 = a >0
```

```
v_0 = b >0
```

```
u_{n+1} = (u_n + v_n)/2
```

```
v_{n+1} = sqrt(u_n * v_n)
```

Dans chacun des cas, on veut une fonction suite(n) qui retourne une liste ou un tableau numpy contenant les n premiers termes de la suite (ou un couple de liste s'il y a plusieurs suites)

La boucle for commence après avoir mis les premiers termes : donc à i=1 pour la première suite (on calcule u1), ou i=2 pour la seconde (on calcule u2).

Dans tous les cas, vous vérifiez au bord de la boucle (i=1 ou 2, i=n-1) : la formule de récurrence ne doit pas faire un appel dans le vide (un u[??] pas encore défini...)

LES FONCTIONS QUI SUIVENT SONT UN EXEMPLE D'IMPLÉMENTATION.  
POUR CE PROBLÈME, TOUTE FONCTION QUI RÉPOND À LA QUESTION EST BONNE.

```
import numpy as np
```

```
def suite1(n):
```

```
    """Calcule les n premiers termes de la suite (u_n), n >= 1
```

```
    u_0 = 5
```

```
    u_{n+1} = 1/(n+1) * exp(-u_n)
```

```
    n -- nombre de termes à calculer
```

```
    retourne une liste u = [u_0, u_1, ..., u_{n-1}]
```

```
    """
```

```
    u = [5] # Création de la liste u, et premier terme : u_0 = 5
```

```
    for i in range(1, n):
```

```
        valeur = 1/i * np.exp(-u[i-1])
```

```
        u.append(valeur)
```

```
    return u
```

```

def suite1n(n):
    """Idem, version numpy"""
    u = np.zeros(n) # Initialisation du tableau qui contiendra (u_n)
    u[0] = 5 # Premier terme : u_0 = 5
    for i in range(1, n):
        u[i] = 1/i * np.exp(-u[i-1])
    return u

## tests
#print(suite1(10))
#print(suite1n(10))

def suite2(n):
    """Suite : n >= 2
    u_0 = 1
    u_1 = 2
    u_{n+2} = sqrt(u_{n+1} + 2*u_n)
    """
    u = [1, 2] # Création de la liste u, et 2 premiers termes : u_0 et u_1
    for i in range(2, n):
        valeur = np.sqrt(u[i-1] + 2*u[i-2])
        u.append(valeur)
    return u

def suite2n(n):
    u = np.zeros(n) # Initialisation du tableau qui contiendra (u_n)
    u[0] = 1 # Premier terme : u_0 = 1
    u[1] = 2 # deuxième terme : u_1 = 2
    for i in range(2, n):
        u[i] = np.sqrt(u[i-1] + 2*u[i-2])
    return u

## tests
#print(suite2(7))
#print(suite2n(7))

def suite3(a, b, n):
    """ Suite :
    u_{n+1} = (u_n + v_n)/2
    v_{n+1} = sqrt(u_n * v_n)
    """
    u = [a]
    v = [b]
    for i in range(1, n):
        valeurU = (u[i-1] + v[i-1]) / 2
        valeurV = np.sqrt(u[i-1] * v[i-1])
        u.append(valeurU)
        v.append(valeurV)
    return u, v

def suite3n(a, b, n):
    u = np.zeros(n)
    v = np.zeros(n)
    u[0] = a
    v[0] = b

```

```
for i in range(1, n):
    u[i] = (u[i-1] + v[i-1]) / 2
    v[i] = np.sqrt(u[i-1] * v[i-1])
return u, v

# tests
print(suite3(1, 11, 7))
print(suite3n(1, 11, 7))
```