P4 - Boucles for et while

Corrigé

Exercice 1 : Afficher 2, 4, 8, 16, 32, 64, ..., 2048.

Il s'agit d'afficher $2^1, 2^2, \dots, 2^{11}$.

In [1]:
```
for k in range(1,12):
    print(2**k)
```

```
2
4
8
16
32
64
128
256
512
1024
2048
```

Exercice 2 : On considère la suite définie par $u_n = \dfrac{10}{n}$ pour $n \geqslant 1$. Afficher les 20 premiers termes de cette suite.

In [2]:
```
for n in range(1,21):
    u = 10/n
    print(u)
```

```
10.0
5.0
3.3333333333333335
2.5
2.0
1.6666666666666667
1.4285714285714286
1.25
1.1111111111111112
1.0
0.9090909090909091
0.8333333333333334
0.7692307692307693
0.7142857142857143
0.6666666666666666
0.625
0.5882352941176471
0.5555555555555556
0.5263157894736842
0.5
```

Exercice 3 : un classique, les suites récurrentes

On considère la suite $(u_n)_{n\in\mathbb{N}}$ définie par $u_0 = 0$ et, pour tout entier naturel n, $u_{n+1} = \sqrt{u_n + 4n}$.

1. Sur feuille, compléter le tableau des calculs successifs à réaliser pour calculer u_3.
2. Écrire un programme qui, étant donné un entier naturel n, calcule et affiche la valeur de u_n.
   On vérifiera que $u_{10} \approx 6.492$.
3. Modifier le programme pour qu'il affiche tous les termes jusqu'à u_n.
4. Déterminer le premier entier naturel n tel que $u_n \geqslant 30$.
   La réponse est $n = 219$.

In [3]:
```
# Question 2
import numpy as np
n = 10
u = 0
for k in range(n):
    u = np.sqrt(u+4*k)
print(u)
```

```
6.492021620534664
```

In [4]:
```
# Question 3
n = 10
u = 0
print(u)
for k in range(n):
    u = np.sqrt(u+4*k)
    print(u)
```

```
0
0.0
2.0
3.1622776601683795
3.893876944661757
4.460255255550041
4.945731013262857
5.380123698695305
5.7775534353820825
6.14634472148952
6.492021620534664
```

In [5]:
```
# Question 4
n = 0
u = 0
while u < 30:
    u = np.sqrt(u+4*n)
    n = n+1
print(n)
```

```
219
```

Exercice 4 : Les compteurs

On considère une urne contenant 8 boules rouges et 5 boules bleues.

1. À l'aide de la bibliothèque numpy.random (à charger), simuler un tirage dans cette urne.

In [6]:
```
import numpy.random as rd
# on numérote les boules : 1 à 8 -> rouge; 9 à 13 -> bleu
alea = rd.randint(1,14)
if alea <= 8:
    print('rouge')
else:
    print('bleu')
```

```
rouge
```

2. Réaliser maintenant 100 tirages dans cette urne et compter le nombre de boules rouges obtenues.

In [7]:
```
compteur = 0  # compteur de boules rouges
for k in range(100):
    # tirage aléatoire
    alea = rd.randint(1,14)
    # On regarde maintenant la couleur
    if alea <= 8:  # si c'est du rouge
        compteur = compteur + 1
print(compteur)
```

```
57
```

In [8]:
```
# Probabilité de tomber sur une boule rouge
8/13
```

Out[8]:
```
0.6153846153846154
```

Donc sur 100 tirages, on a en moyenne 62 boules rouges. Cela se vérifie.

Exercice 5 : Suites récurrentes, bis

Soit $v_1 = 2$ et $v_{n+1} = \exp\left(\frac{v_n}{n+1}\right)$ pour $n \geqslant 1$.

1. Afficher les 20 premiers termes. Conjecturer la limite de (v_n).
2. Déterminer le premier $n \in \mathbb{N}$ tel que $v_n \leqslant 1.01$. *On trouve 102.*
3. Déterminer le plus grand $n \in \mathbb{N}$ tel que $v_n \geqslant 1.005$. *On trouve 201.*

In [9]:
```
# Question 1
v = 2
print(v)
for k in range(1,20):
    v = np.exp(v/(k+1))
    print(v)
```

```
2
2.718281828459045
2.4746375548688673
1.8564376116619543
1.4495998057910315
1.2732847358151316
1.1994916228466597
1.1617604138259945
1.1377862521488562
1.120504046083549
1.1072328824550528
1.096660230087974
1.0880188560779138
1.080815266313941
1.074713754337871
1.06947685638356
1.0649314206335223
1.0609480091651493
1.0574278143044635
1.0542940443967925
```

La suite semble tendre vers 1

In [10]:
```
# Question 2
n = 1
v = 2
while v > 1.01:
    v = np.exp(v/(n+1))
    n = n+1
print(n)
```

```
102
```

In [11]:
```
# Question 3
n = 1
v = 2
while v >= 1.005:
    v = np.exp(v/(n+1))
    n = n+1
print(n-1)
```

```
201
```

Exercice 6 : Suite récurrente d'ordre 2

Soit $(u_n)_{n\in\mathbb{N}}$ la suite définie par

$$\begin{cases} u_0 = 1,\ u_1 = 2 \\ \forall n \in \mathbb{N},\ \ u_{n+2} = 2u_n - u_{n+1} \end{cases}$$

Écrire un programme python qui, étant donné un entier naturel n, renvoie la valeur de u_n. $u_{20} = -349524$.

In [12]:
```
n = 20
u = 1 # u0
v = 2 # u1
for k in range(n):
    w = 2*u-v  # u(k+2)
    u = v
    v = w
print(u)
```

```
-349524
```

Exercice 7

On appelle suite de Syracuse toute suite d'entiers naturels $(u_n)_{n\in\mathbb{N}}$ définie de la manière suivante : $u_0 \in \mathbb{N}^*$ et pour tout entier naturel n :

$$u_{n+1} = \begin{cases} \dfrac{u_n}{2} & \text{si } u_n \text{ est pair;} \\ 3u_n + 1 & \text{si } u_n \text{ est impair.} \end{cases}$$

1. La division euclidienne peut-être obtenue avec les commandes suivantes : a // b : quotient a % b : reste En déduire une façon de tester si un entier n est pair (on écrira un booléen).
2. Afficher les 25 premiers termes de la suite en partant de $u_0 = 15$.

In [13]:
```
n = 16
n%2 == 0
```

Out[13]:
```
True
```

In [14]:
```
u = 15
print(u)
for k in range(24):
    if u%2 == 0:
        u = u/2
    else:
        u = 3*u+1
    print(u)
```

```
15
46
23.0
70.0
35.0
106.0
53.0
160.0
80.0
40.0
20.0
10.0
5.0
16.0
8.0
4.0
2.0
1.0
4.0
2.0
1.0
4.0
2.0
1.0
4.0
```

Exercice 8 : programmons un petit jeu !

1. Le jeu que nous allons programmer demande aux joueurs d'intervenir en cours de partie pour donner à l'ordinateur une valeur. Pour cela, nous allons utiliser la

fonction **input**.

- Instruction : input(prompt)
- Paramètre : prompt, une chaîne de caractères représentant un message à afficher avant la saisie
- Sortie : une chaîne de caractères

Exécuter les cellules suivantes pour comprendre le fonctionnement de input.

In [15]:
```
x = input('Quel est votre prénom ? ')
print('Bonjour ' + x)
```

```
Quel est votre prénom ? Cécile
Bonjour Cécile
```

In [16]:
```
valeur = input('Saisissez un nombre : ')
print(valeur)
print(type(valeur))
```

```
Saisissez un nombre : 5
5
<class 'str'>
```

In [17]:
```
valeur = valeur + 1
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-17-939af331f4fc> in <module>
----> 1 valeur = valeur + 1

TypeError: can only concatenate str (not "int") to str
```

La sortie d'un input est une chaîne de caractères, même si l'utilisateur a saisi un nombre.

Pour pouvoir manipuler la valeur saisie comme un nombre, il faut la *convertir*.

In [18]:
```
valeur = input('Saisissez une valeur réelle : ')
valeur = float(valeur)
print(valeur)
print(type(valeur))
```

```
Saisissez une valeur réelle : 2.4
2.4
<class 'float'>
```

In [19]:
```
valeur = valeur + 1
```

On peut aussi convertir en *int* si la valeur à saisir est entière.

In [20]:
```
valeur = input('Saisissez une valeur entière : ')
valeur = int(valeur)
print(valeur)
print(type(valeur))
```

```
Saisissez une valeur entière : 5
5
<class 'int'>
```

Exercice 8 (suite)

On souhaite programmer le jeu de Nim (ou jeu des bâtons). Au début, on dispose de 20 bâtons alignés. Les deux joueurs vont, à tour de rôle, enlever 1, 2 ou 3 bâtons. Le joueur qui laisse 1 bâton à la fin a gagné (autrement dit, celui qui prend le dernier bâton a perdu).

2. On définit une variable **joueur** égale à 1 ou 2 (numéro du joueur dont c'est le tour). Écrire une instrution d'une ligne permettant de passer d'un joueur à l'autre :
   - si joueur = 1, le résultat doit être joueur = 2;
   - si joueur = 2, le résultat doit être joueur = 1.
3. Programmer le jeu. À chaque tour, on affichera 'Joueur 1' ou 'Joueur 2' puis on lui demandera de saisir le nombre de bâtons qu'il souhaite enlever et enfin le nombre de bâtons restant. Le jeu s'arrête quand il reste un seul bâton. On affichera le numéro du gagnant.

In [21]:
```
joueur = 2
joueur = 3 - joueur
joueur
```

Out[21]: 1

In [23]:
```
batons = 20  # nombre de bâtons
joueur = 1
while batons > 1:
    print('Il reste', batons, 'batons.')
    print('Joueur', joueur)
    prise = int(input('Combien prenez-vous de batons ? '))
    batons = batons - prise
    joueur = 3 - joueur

print('Il reste ' + str(batons) + ' baton.')
joueur = 3 - joueur
print('Le gagnant est le joueur', joueur)
```

```
Il reste 20 batons.
Joueur 1
Combien prenez-vous de batons ? 3
Il reste 17 batons.
Joueur 2
Combien prenez-vous de batons ? 2
Il reste 15 batons.
Joueur 1
```

```
Combien prenez-vous de batons ? 2
Il reste 13 batons.
Joueur 2
Combien prenez-vous de batons ? 1
Il reste 12 batons.
Joueur 1
Combien prenez-vous de batons ? 3
Il reste 9 batons.
Joueur 2
Combien prenez-vous de batons ? 3
Il reste 6 batons.
Joueur 1
Combien prenez-vous de batons ? 1
Il reste 5 batons.
Joueur 2
Combien prenez-vous de batons ? 2
Il reste 3 batons.
Joueur 1
Combien prenez-vous de batons ? 2
Il reste 1 baton.
Le gagnant est le joueur 1
```