# Exercice Python itératif

1. Faire une table (liste de liste) de 10 par 10 avec le résultat de la multiplication.

1 2 3 4 …

2 4 6 8 …

3 6 9 12 …

… … …

10 20 30 40 ...

def initTable():

    liste=[]

    for i in range(1,11):

        liste.append(list(range(0,11\*i,i)))

    return liste

1. Écrire une fonction fact de calcul de factorielle en programmation itérative.

fact(5) = 5 \* 4 \* 3 \* 2 \* 1

print(fact(5)) → 120

def fact (n):

    result=1

    for i in range(1,n+1):

        result\*=i

    return result

1. Écrire une fonction init d’initialisation d'une liste de taille n avec des 0

print(init(5)) → [0,0,0,0,0]

def init (n):

    liste=[]

    for i in range(0,n):

        liste.append(0)

    return liste

1. Écrire une fonction de recherche du plus grand élément d’une liste en programmation itérative.

print(recherche([2,7,9,4]) → 9

def rechercheMax (liste):

    pge=liste[0]

    for i in liste:

        if (i>pge):

            pge=i

    return pge

1. Écrire une fonction moyenne qui renvoie la moyenne des éléments d’une liste.

print(moyenne([2,4,6,8])) → 5

def moyenne(liste):

    somme=0

    for i in liste:

        somme+=i

    return somme/len(liste)

1. Écrire une fonction contain qui vérifie si une fonction contient ou non l’élément demandé.

print(contain([0,2,4,8],2) → True

print(contain([0,2,4,8],3) → False

def contain(liste, elem):

    for i in liste:

        if i == elem:

            return True

    return False

1. Écrire une fonction occurs qui compte le nombre d'occurrence (d’apparition) d’un élément dans une liste.

print(occurs([2,5,4,2,6,7,8,5,7],7) → 2

print(occurs([2,5,4,2,6,7,8,5,7],4) → 1

print(occurs([2,5,4,2,6,7,8,5,7],1) → 0

1. Écrire une fonction recherchePlusPresent qui recherche l’élément qui est présent un le plus grand nombre de fois dans une liste.

RecherchePlusPresent([1,4,5,2,1,4,5,2,1,2,,5,6,8,7,4,2,4,1,2,2,12,4,5,2,1,2]) → 2

1. Écrire une fonction fusionListes qui fusionne deux listes triées.

print(fusion([2,4,5,6,8],[1,3,7]) –> [1,2,3,4,5,6,7,8]

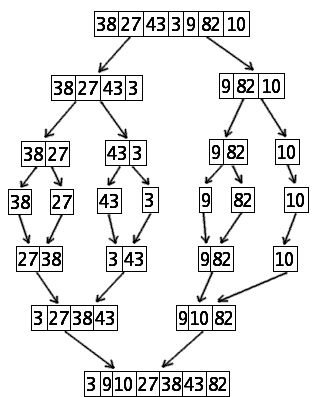
1. Ecrire une fonction contains qui renvoi vrai si tous les éléments de la liste 2 sont contenues dans la première, faux sinon;

print(contains([2,5,7,9,6],[2,6]) → vrai

1. Fonction swap qui échange le premier et le dernier élément d’une liste.

print(swap([1,5,8,10])) → [10,5,8,1]

1. Algorithme de tri par fusion de deux tableaux (https://fr.wikipedia.org/wiki/Tri\_fusion)



# Exercice Python récursif

1. Écrire une fonction de calcul de factorielle en programmation récursive.

fact(5) = 5 \* fact(4)

1. Écrire une fonction de recherche du plus grand élément d’une liste en programmation récursive.
2. Écrire une fonction récursive qui permettent de calculer la longueur d’une liste sans utiliser la fonction len().

# Exercice avancé

1. **GuessMyNumber :**

Donner une borne inf et sum. L’algorithme choisit un nombre que le joueur doit deviner. L’algorithme dit alors si le nombre proposé est plus ou moins grand que le nombre choisi.

Optionnel : Affiche le nb d’étapes / Afficher le temps

1. **Syracuse**
2. Écrire une fonction syracuse, qui pour un entier n, donne le nombre d’étapes pour arriver à 1 en respectant l’algorithme suivant :

Si le nombre est pair, on le divise par 2

Si le nombre est impaire, on le multiplie par 3 et on ajoute 1 (n\*3+1)

Ainsi, pour 10 🡪 5 🡪 16 🡪 8 🡪 4 🡪 2 🡪 1

Donc syracuse(10) donne 6 car il y a 6 étapes pour arriver à 1.

1. Faire la liste des résultats donnés par cette algo sur les 10000 premiers entiers.
2. Quel est le nombre qui a donné le plus grand nombre de de cette liste
3. Combien de temps l’algo a-t-il mis pour calculer pour les 10000 premiers entiers ?
4. Comment pourrais t’on améliorer ce résultat.
5. **Planche de galton**



Simuler le résultat d’une planche de galton (de 9 de largeur et comprenant 10 étages) sur 1000 lancers.

A chaque étape, la balle dévie de 1 sur la gauche ou sur la droite (il y a donc ici 10 étapes).

1. Simuler le jeu de la vie sur un tableau de 100 par 100.

<https://www.youtube.com/watch?v=S-W0NX97DB0>

# Cours :

Chaînes youtube :

CodeAvecJonathan : <https://www.youtube.com/watch?v=oUJolR5bX6g>

Docstring python

Cours écrit avec exercice mais parfois un peu complexe : https://python.sdv.univ-paris-diderot.fr/cours-python.pdf

# Jeux :

Leekwars

CodinGame

Codewars

# Exercices :

Pour tout niveau, mais pas toujours simple :

<http://www.jaicompris.com/python/python-exercice.php>

<https://developpement-informatique.com/cat/16/langage-python>

# Projet

https://www.python-lycee.com/activite-en-ligne-mathematiques