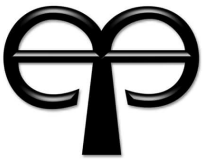

TP 1 : Bases de Python (2018-2019)
Ndeye Fatou NGOM

Exercice 1 *Généralité sur les variables*

1. *Environnement de travail*
   (a) *Lancer le navigateur Anaconda,*
   (b) *Choisir l'IDE Spyder et la console Ipython.*
2. *Créer un nouveau dossier Test Python et changer le répertoire de travail avec l'une des méthodes qui suivent*
   (a) ***pwd***, ***cd***

```
pwd #-> 'D:\\Python\\NotesCoursNFN2018'
cd D:\Python\NotesCoursNFN2018
D:\Python\NotesCoursNFN2018
pwd #-> 'D:\\Python\\NotesCoursNFN2018'
```

   (b) ***module os***, *fonctions* ***getcwd*** *et* ***chdir***

```
import os
os.getcwd() #-> 'D:\\Python\\NotesCoursNFN2018'
os.chdir('D:\Python\ScipyNoteCours')
os.getcwd() #-> 'D:\\Python\\ScipyNoteCours'
```

Documentation

1. *Executer pas à pas les commandes*

```
print ?
print?
?print
help(print)
```

2. *Qu'observez-vous ?*
3. *Lister les mots clés réservés de Python à l'aide de help("keywords"),*

Donner les types des variables qui suivent à l'aide de la fonction type() ou de la commande %whose

```
a = 5
d=floa(a)
b = 5.0
c = 0.1 + 5j
c.real
c.imag
t = (a > 7)
```

Executer pas à pas les opérations

```
7 * 3
2**10
2^10
8 % 3
7/2
7//2
a=c=d=e=4
a,c,d,e=1,2,3,4
a,b=b,a
```

Qu'observez voous ?

Examiner les codes qui suivent

1. *Chaine de caractères*
   (a) *Exécuter ligne par ligne le script dans la console et commenter le résultat*

   ```
   C= str(input("text: "))
   print(C)
   type(C)
   l=len(s)
   print(l)
   ```

      i. *Compter le nombre d'apparition du caractère a,*
      ii. *Convertir en majuscule,*
      iii. *Qu'observez avec les caractères accentués.*

   (b) *Observer le résultat en ajoutant au début du script*

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

   (c) *Commenter les lignes de codes qui suivent*

   ```
   c=C[:7]
   print(c)
   c[0]="A"
   id=c.find("a")
   print(id)
   n=s.count("a")
   print(n)
   s2=s.replace("a","b")
   ```

   (d) *Commenter les lignes qui suivent*

   ```
   L=list(C)
   type(L)
   print(L)
   C2=C.split(" ")
   type(L)
   ```

(e) *Définir la liste constituée des étudiants de la classe puis*

i. *affichez la liste des étudiants,*

ii. *Ajoutez un nouveau étudiant Marie à la liste et affichez la liste ;*

iii. *Inversez et affichez la liste ;*

iv. *affichez l'indice de l'étudiant X sur la liste ;*

v. *enlevez l'étudiant Y et affichez la liste ;*

vi. *affichez la sous-liste du 1 eau 5 étudiant ;*

vii. *affichez la sous-liste du début au 3 élément ;*

viii. *affichez la sous-liste du 2 élément à la fin de la liste ;*

ix. *affichez la sous-liste des étudiantes de la liste ;*

(f) *Faite la même chose cette fois ci en associant à chaque étudiant son âge.*

2. *Tester et interpréter le segment de code qui*

```
from turtle import *
n=4; c=15
a=360.0/n
    while n>0:
        forward(c)
        left(a)
        n=n-1
```

3. *Modifier le manière à générer des formes plus complexes à l'aide des structures de contrôles de flux.*

Exercice 2 *Généralité sur les variables*

1. *Executer la commande suivante*

```
help ("keywords")
```

Qu'observez vous

2. *Utiliser les fonctions d'un module*

(a) *Après avoir importé le module math, tester certaines de ces fonctions décrites à l'adresse suivante*

```
https://docs.python.org/fr/3.5/library/math.html
```

(b) *Tester et interpréter le segment de code qui*

```
# console
%gui tk
#Editeur
from turtle import *
```

```
n=4; c=15
a=360.0/n
    while n>0:
        forward(c)
        left(a)
        n=n-1
```

(c) *Modifier le script de manière à générer des formes plus complexes à l'aide des structures de contrôles de flux et des fonctions qui suivent*

```
reset() #effacer tout et recommencer
goto() # aller au point de coordonnées (x,y)
forward(d) #avancer d'une distance d
backward (d) #reculer d'une distance d
up() # relever le crayon (on ne dessine plus)
down() #abaisser le crayon (on recommence à dessiner)
color(couleur) #definir une couleur
left(angle) #tourner à gauche d'un angle donné
right(angle) #tourner à droite d'un angle donné
width(epaisseur) #choisir l'épaisseur du tracé
....
```

Pour plus de details sur ce module

```
https://docs.python.org/2/library/turtle.html
```

3. *Decriver les scripts qui suivent et tester les pour les valeurs, 5, 100, 1010.*

```
#cas non recursive sans module math
def fact1(x):
    x=int(x)
    r=1
    while(x>1):
        r=r*x
        x=x-1
    return r

#cas 2 recursive
def fact2(n):
    if n<2:
        return 1
    else:
        return n*fact2(n-1)
```

Q'observez vous ?

4. *Fonctions sur les chaines*

(a) *Executer et interpreter les segments de codes qui suivent*

```
C='Fatou va au marche'
#split
L=C.split()
C.split('a')
#join
' '.join(L)
#find
print(C.find('va'))
#count
print(C.count('va'))

#lower(),upper(),title(),swapcase()
C.lower()
C.upper()
C.title()
print(C.swapcase())
#replace
print(C.replace('va','ou'))
#index
print(C.index('ou'))
```

(b) *Definir une fonction prenant en paramètre deux chaines de caractères (mot et texte) et renvoie True si le mot apparaît dans le texte,*

(c) *Tester et interpreter le segment de code qui suit*

```
def compterMots(texte):
    dict = {}
    listeMots = texte.split()
    for mot in listeMots:
        if mot in dict:
            dict[mot] = dict[mot] + 1
        else:
            dict[mot] = 1
        return dict
```

5. *Les fichiers*

(a) *Modes d'ouverture d'un fichier*

```
Ouverture
fichier = open("fichier.txt", "mode")
#mode=
r #ouverture en lecture (READ).
w #ouverture en écriture (WRITE)
# à chaque ouverture le contenu du fichier est écrasé.
# Si le fichier n'existe pas python le crée.
a #pour une ouverture en mode ajout à la fin du fichier (APPEND).
# Si le fichier n'existe pas python le crée.
b #une ouverture en mode binaire.
t #ouverture en mode texte.
```

```
x #crée un nouveau fichier et l'ouvre pour écriture
Fermeture
fichier.close()
```

(b) *Créer un fichier fichier.txt où vous mettrez du texte et ensuite tester et interpreter la fonction qui suit*

```
def rechercherMotFichier(chaine, "fichier.text"):
    chaine="mot"
    fichier=open("fichier.txt","r")
    for ligne in fichier:
        if chaine in ligne:
            print ligne
            fichier.close()
```

(c) *Définir une fonction qui permet de rechercher plusieurs motifs dans un fichier.*

6. *Fonctions récursives*

(a) *Dessiner sur un papier la fractale dite de Von Koch en considérant les étapes suivantes*

i. *En partant d'un segment de droite initial,*

ii. *On divise le segment de droite en trois segments, de longueurs égales,*

iii. *On construit un triangle équilatéral ayant pour base le segment médian de la première étape*

iv. *Selon le nombre d'itérations demandé, on réitère ce processus pour chaque segment.*

(b) *Tester et interpréter les fonctions qui suivent*

```
from turtle import*
def decale_vers_la_gauche(largeur):
    left(180)
    penup()
    forward(largeur)
    pendown()
    left(180)

def decale_vers_le_haut(hauteur):
    left(90)
    penup()
    forward(hauteur)
    pendown()
    left(90)
```

(c) *Executer le script qui suit*

```
def koch_n(n,longueur):
    speed(0) #vitesse du traceur max
    pencolor("blue") #couleur du stylo
    shape("turtle") #forme
    if n==0:
        forward(longueur)
    else:
        koch_n(n-1,longueur/3)
        left(60)
        koch_n(n-1,longueur/3)
        right(120)
        koch_n(n-1,longueur/3)
        left(60)
        koch_n(n-1,longueur/3)
decale_vers_la_gauche(300)
decale_vers_la_gauche(-200)

koch_n(4,300)
```

Qu'observez vous.
Pour plus de details sur les structures fractales referez vous aux adresses suivantes

```
https://www.lespritsorcier.org/blogs-membres/module-turtle-de-python
http://natureofcode.com/book/chapter-8-fractals/
```

(d) *Modifier le script de manière à générer un autre fratale de votre choix.*

Exercice 3 *Programmation modulaire*
Nous allons explorer la programmation modulaire dans cette partie à l'aide de tkinter de la bibliothèque standard et des automates cellulaires. Pour chacun des exemples, créer une classe qui contient une grille du modèle et les méthodes qui permettront son évolution.

1. *Jeu de la vie*
   (a) *Le jeu de la vie repose sur le principe d'évolution d'une grille 2D dans le temps. A chaque étape, appelée génération, les cellules évoluent en fonction de leur voisinage. Chaque cellule a 8 cellules voisines.*
   (b) *Regles*
      i. *Si une cellule vivante est trop isolée (0 ou 1 voisin), alors elle meurt à l'évolution suivante,*
      ii. *Si elle est raisonnablement entourée (2 ou 3 voisins), alors elle reste en vie,*
      iii. *Si elle est entourée de trop de cellules (4 voisins ou plus), elle meurt à la génération suivante (mort par sur population)*
      iv. *Si une cellule morte est entourée de 3 cellules vivantes, alors elle devient vivante à la prochaine évolution (naissance par reproduction)*
   (c) *Telecharger l'implementation proposée à l'adresse*

      `http://www.nymphomath.ch/pj/automates/jeudelavie.py`

   (d) *Interprétez les différentes fonctions intervenant dans la construction du programme*
   (e) *Améliorer ces modèles en y intégrant d'autres variables et des hypothèses supplémentaires.*