CHAPITRE : 1

PREMIERES NOTIONS DE PYTHON sous PYZO

1.1 / Paramétrage rapide de l'environnement pyzo

Cliquer sur 'Settings' puis sur 'Select language' afin de choisir la langue française.
Dans 'Affichage' choisir zoom afin d'agrandir la police de caractères.

1.2 / Le mode interactif

(ou mode calculatrice)

Le dialogue s'effectue dans la fenêtre supérieure.
Les ordres se tapent après la commande In[i]:
Les réponses se lisent après la commande Out[i]:

Dans la fenêtre supérieure, après In[1]: *taper*

*2*3 puis presser la touche entrée*
*7+3*4*
20/3
20//3
a=7
a
b=a
b
nom='truc'
nom
len(nom)
print(nom[0],'*',nom[1],'*',nom[2],'*',nom[3])
print(nom[0:3])
print(nom[1:3])
x=y=8
x
y
a,b=2,3.4
a
b
print(a,nom,b)
print(type(a),type(nom),type(b))
*2**4*
*1000**1000*
*(-1)**(1/2)*

.

1.3 / Le mode programmation

1.3.1 Ecriture d'un programme (ou script)

Choisir : *Fichier / Nouveau.*
Cela ouvre une fenêtre d'édition pour saisir les lignes de codes du programme.
L'onglet de cette fenêtre d'édition du code porte alors le nom < tmp$i$ >
Taper successivement chaque ligne de code.
Une ligne commençant par # ne sera pas exécutée, elle contiendra des commentaires.

1.3.2 Première sauvegarde d'un programme

Faire un clic droit sur l'onglet < tmp$i$ > ou choisir *Fichier / Enregistrer sous*
Choisir alors le répertoire qui doit contenir le programme puis nommer
le programme à enregistrer et valider.
Attention :
Les noms de fichiers ne doivent contenir que des lettres ou des chiffres.
Pas d'espace dans les noms de fichier.
L'extension " *py* " sera automatiquement attribuée. Elle signifie fichier Python.
Après chaque modification du programme l'onglet sera précédé d'une icône rouge.
Cela signifie que la dernière version du programme n'a pas été sauvegardée.
Il faut alors, avant d'exécuter le script, en sauvegarder la dernière version.

1.3.3 Sauvegardes successives d'un programme

Choisir *Fichier / Enregistrer* ou alors presser simmultanément $CTRL$ et S

1.3.4 Exécution d'un programme

Si le programme à exécuter est déjà ouvert dans la fenêtre d'édition :
Choisir *Exécuter / Démarrer le script* ou presser $F5$ ou encore cliquer sur l'onglet du programme et choisir *Exécuter.*
La dernière version du programme est alors automatiquement conservée.
L'exécution s'effectue alors dans la fenêtre supérieure.

Si le programme à exécuter n'est pas déjà ouvert dans la fenêtre d'édition :
Dans la fenêtre principale choisir *Fichier / Ouvrir* et aller chercher le fichier dans le répertoire qui le contient.
Choisir *Exécuter / Démarrer le script* ou presser $F5$ ou encore cliquer sur l'onglet du programme et choisir *Exécuter.*
La dernière version du programme est alors automatiquement conservée.
L'exécution s'effectue alors dans la fenêtre supérieure.

CHAPITRE : 2

LES ENTREES SORTIES EN PYTHON sous PYZO

2.1 / Entrée sortie de variables numériques

2.1.1 Premier script ou programme

Voici un programme qui demande l'âge de l'utilisateur puis qui affiche cet âge

```
# exo2111 entrée sortie d'une var numérique
print('entrer votre âge ')
age=input()
print('vous avez ',age,' ans')
```

Ecrire ce programme.
Le sauvegarder dans un répertoire précis sous le nom "exo2111"
Ce fichier portera désormais le nom "exo2111.py"
Demander l'exécution de ce programme.
Le programme vous demande alors de saisir votre âge . A la fin de la saisie presser entrée

Afin de comprendre les instructions de ce programme :

La ligne qui suit le symbole # est du commentaire et n'est pas éxécutée.
La commande print est un ordre d'affichage.
'entrer votre age ' est un commentaire d'invitation à la saisie.
age est une variable qui contiendra le résultat de la saisie.

Exercice2112

Exécuter le programme suivant et ... comprendre !

```
# exo2112 entrée sortie d'une var numérique
a=input('entrer votre âge ')
print(a)
print(type(a))
b=float(a)
print(b)
print(type(b))
```

Attention, en Python la fonction *input* renvoie toujours une variable de type chaine de caractères. On peut ensuite transfrormer cette valeur en nombre entier avec l'instruction *int* ou en nombre réel avec l'instruction $float$

Exercice2113

Ecrire un programme nommé exo2113 qui saisit un nombre x puis qui affiche le double de *ce nombre*

2.1.2 Exemple de programme

Comprendre chacune des instructions du programme suivant

```
# exo212
x=input('donner x ');
y=input('donner y ');
z=y;
y=x
x=z
print('x= ',x)
print('y= ',y)
```

2.2 / Entrée sortie d'une variable chaine de caractères

2.2.1 Premier exemple

Voici un programme qui demande le prénom puis le nom de l'utilisateur puis qui affiche, par concaténation, le nom suivi du prénom.

```
# exo221 entrée sortie d'une var chaine de caractères
prenom=input('entrer votre prénom ');
print(prenom)
prenom=input('entrer votre nom ');
print(nom)
complet=nom+'  '+prenom
print(complet)
```

2.2.2 Exemple d'affichage de plusieurs zones de texte

```
# exo222 entrée sortie de deux variables
print('entrer votre prénom ')
prenom=input()
print('entrer votre age ')
age=input();
print('bonjour ',prenom,' tu as ',age,' ans ')
```

.

2.2.3 Arrondi d'une variable numérique

Exercice223

Taper le programme ci-dessous .

```
# exo223 arrondi d'une var numérique
from math import *
a=input('entrer une valeur décimale ')
a=float(a)
print('floor(',a,') =' ,floor(a))
print('int(',a,') = ',int(a))
print('round(',a,') = ',round(a))
```

La ligne "**from math import ***" sigifie que les instructions qui la suivent nécessitent l'utilisation du module (ou bibiothèque) de mathématiques.

L'éxécuter pour $x \in \{0.4 \;\; 0.5 \;\; 0.6 \;\; -0.4 \;\; -0.5 \;\; -0.6 \;\; -1.4 \;\; -1.5 \;\; -1.6 \;\; 3 \;\; 4\}$
Résumer les résultats dans un tableau contenant en colonnes les 11 *valeurs de* x *et en lignes les résultats des appels des fonctions floor, int et round pour chaque valeur de* x.
Quel est le rôle de chaque fonction ?
Représenter chacune de ces fonctions sur un graphique différent.

2.2.4 Variables complexes

Taper le programme suivant et l'exécuter:

```
# exo224 travaux sur les nombres complexes
from math import *
x=1+2*1j
print('x=',x)
z=2*x
print('z=',z)
print('|z|=',abs(z))
print('Re(z)=',z.real)
print('Im(z)=',z.imag)
```

CHAPITRE : 3

LES STRUCTURES CONDITIONNELLES

ET LES VARIABLES BOOLEENNES

en PYTHON sous Pyzo

3.1 / La structure IF

if condition **:**
action 1
action 2
.....
action n

Attention en langage Python, *if condition : s'appelle un ligne d'en-tête*
Les actions comprises dans la structure conditionnelles sont automatiquement indentées
Il n'y a pas de "*end*".

Exercice311

En s'inspirant de l'exercice 212 , écrire un programme nommé exo311 qui saisit la moyenne obtenue au BAC par un candidat puis qui affiche éventuellement le message " candidat reçu "

Exercice312

Ecrire un programme nommé exo312 qui saisit la moyenne obtenue au BAC par un candidat puis qui affiche le message " candidat reçu " ou le message " candidat refusé "

3.2 / Les opérateurs de comparaison

Les opérateurs de comparaison fréquemment utilisés sont :

== (égalité) ; ! = (différent) ; < ; > ; <= (inf ou égal) ; >=

syntaxe :

```
if a == b:
    action
```

```
if a!= b:
    action
```

```
if a <= b:
    action
```

3.3 / La structure IF ELSE

if condition **:**
    action 1
    action 2
else :
    action 3
    action 4
    action 5

Exercice331

Reprendre l'exercice 312 à l'aide de la structure if else end afin de minimiser le nombre de questions

3.4 / Les opérateurs logiques

Les opérateurs logiques fréquemment utilisés sont :

and (et) ; *or* (ou inclusif) ; *not* (négation)

syntaxe :

if not(cond1) : action	if cond1 *and* cond2 : action	if cond1 *or* cond2 action

3.5 / Les variables booléennes

On appelle variable booléenne (ou variable logique) une variable qui ne peut prendre que deux valeurs : $True$ ou $False$

Exercice351

Soit n un nombre entier.
Soit D2 une variable prenant la valeur vraie ou faux selon que n est multiple de 2
Soit D3 une variable prenant la valeur vraie ou faux selon que n est multiple de 3
Ecrire un algorithme qui saisit n et affiche les messages suivants s'ils sont opportuns:
" n est un multiple de 6 "
" n est multiple soit de 2 soit de 3 "
" n est multiple de 2 ou de 3 "
" n n'est multiple ni de 2 ni de 3 "
" n est multiple de 2 mais pas de 3 "
" n est multiple de 3 mais pas de 2 "
Programmer cet algorithme en Python sans utiliser la structure else

Exercice352

Programmer l'algorithme précédent en Python en utilisant la structure else

Exercice353

Ecrire un programme qui saisit la moyenne obtenue au BAC et qui attribue une éventuelle mention.
Penser à utiliser une variable booléenne

CHAPITRE : 4

LES FONCTIONS EN PYTHON 3.3 sous Pyzo

4.1 / Fonctions avec transmission de paramètres en entrée

Pour son éxécution, la fonction nécessite la donnée de p valeurs en entrée e_1, e_2, ... , e_p .

4.1.1 Exemple

```
#exo411 exemple de fct: quotient de 2 nombres
def div(a,b):
    q=a/b
    print(q)
```

Ecrire le fichier exo411 dans la fenêtre d'édition puis le sauvegarder.
Exécuter le fichier 411 puis dans la fenêtre principale taper ensuite : div$(8, 4)$.
L'éxécution de la fonction div nécessite l'entrée de deux paramètres a et b
Il s'agit d'une **transmission de valeurs à l'entrée**
On obtient alors l'appel de la fonction div pour la valeur de $a = 8$ et $b = 4$
La réponse est alors 2.

4.1.2 Exemple

```
#exo412 exemple de fct
def f(a,b):
        x=a+b
        y=a-b
        print(x,y)
```

L'éxécution de la fonction f nécessite l'entrée de deux paramètres a et b
L'appel $f(5, 9)$ affichera 14 et -4

Exercice413

Recharger en mémoire l'exercice 353, le renommer exo413 puis le modifier afin d'utiliser une fonction bac1 pour entrer la moyenne au BAC et afficher élève reçu ou refusé.

Exercice414

Ecrire un algorithme qui détermine l'obtention du BAC avec une éventuelle mention

Recharger en mémoire l'exercice 413, le renommer exo414. Le modifier afin d'utiliser une fonction bac2 de deux variables pour entrer le nom du candidat puis la moyenne au BAC. Après analyse, afficher un message du genre : candidat DUPONT reçu

Exercice415

Ecrire un algorithme qui détermine les éventuelles solutions de l'équation $ax + b = 0$
Programmer cet algorithme à l'aide d'une fonction resol1 de deux variables, a et b.
Imaginer un jeu d'essais

Exercice416

Ecrire un algorithme qui détermine les éventuelles solutions réelles de l'équation $ax^2 + bx + c = 0$.
Si $a = 0$, *appeler la fonction resol1 de l'exercice précédent.*
Programmer cet algorithme à l'aide d'une fonction resol2 appelant éventuellement la fonction resol1 qui sera à placer avant la fonction resol2.
Imaginer un jeu d'essais.

Exercice417

Reprendre l'exercice 353 et 414 puis les modifier afin d'afficher un commentaire du style : candidat DUPONT reçu mention.très bien.
Utiliser une fonction bac3 ainsi que des instructions elif.

4.2 / Fonctions avec transmission de paramètres en sortie

La fonction f ne nécessite aucune valeur en entrée.
Après éxécution, cette fonction f restitue n valeurs de sortie : s_1, s_2, ... , s_n
à l'aide de l'instruction *return*

```
#exo421
def note() :
    n1=input(' entrer la première note ')
    n2=input(' entrer la deuxième note ')
    print('n1=',n1)
    print('n2=',n2)
    return (n1,n2)
```

Lors de l'éxécution de la fonction note deux valeurs sont saisies
Ces deux valeurs seront transmises vers un autre programme,
Il s'agit de **transmission de valeurs en sortie** avec l'instruction *return*.

4.3 / Fonctions avec transmission de paramètres en entrée et en sortie

Pour son éxécution, la fonction f nécessite la donnée de p valeurs en entrée e_1, e_2, ... , e_p .
Après exécution, cette fonction restitue n valeurs de sortie : s_1, s_2, ... , s_n avec l'instruction *return*.

```
# exo431 fonction avec paramètres d'entrée et de sortie
from math import *
# fct d'affichage ou de sortie
def affich(t,u,v,w):
    print('la moy arithm est ',t)
    print('la moy géométrique est ',u)
    print('la moy quadratique est ',v)
    print('la moy harmonique est ',w)
# fct de calcul interne
def calc(x,y):
    a =(x+y)/2
    b=sqrt(x*y)
    c=sqrt((x**2+y**2)/2)
    d=2/(1/x+1/y)
    return(a,b,c,d)
# fct de saisie des informations
def lect():
    n1=input('entrer la première note')
    n2=input('entrer la seconde note')
    n1=float(n1)
    n2=float(n2)
    return(n1,n2)
# programme principal
(x,y)=lect()
(moy1,moy2,moy3,moy4)=calc(x,y)
affich(moy1,moy2,moy3,moy4)
```

Explications :

Le programme écrit ci-dessus comporte trois fonctions distinctes dont les noms sont : **lect, calc et affich.**
Le programme principal gère l'appel des trois fonctions .

- (x,y)=lect() appelle la fonction **lect** .
  - Cette fonction **lect** saisit deux notes, n1 et n2 qui sont transmises en paramètres de sortie vers le pg principal qui les nomme x et y.
- (moy1,moy2,moy3,moy4)=calc(x,y) appelle la fonction **calc** à l'aide des paramètres d'entrée x et y.
  - La fonction **calc** calcule quatre moyennes a, b, c et d qui sont transmises comme paramètres de sortie vers le programme principal qui les nomera moy1, moy2, moy3 et moy4.
- affich(moy1,moy2,moy3,moy4) appelle la fonction **affich** à l'aide des paramètres d'entrée moy1, moy2, moy3 et moy4 .
  - dans **affich**.les variables t, u, v et w prennent respectivement en entrée les valeurs de moy1, moy2, moy3 et moy4 puis l'affichage est exécuté.

.

Exercice432

Reprendre l'exercice 417 puis le modifier selon le schéma de l'exercice 431.

- La fonction **lect** saisissant *nom* et *moy*.
- La fonction **calc**, appelé avec *moy*, retourne *statut*=(reçu ou refusé) ainsi que *mention=('AB '... 'TB', 'félicitations .. ').*
- La fonction **affich** restituant *nom*, *statut* et *mention.*

4.4 / Notion de variables locales ou globales

4.4.1 Variables locales

Dans une fonction, tout paramètre d'entrée ou de sortie est une variable locale.
Dans une fonction, toute variable interne à la fonction est dite locale.
Si, dans une fonction, une variable a est locale alors on ne peut pas utiliser la valeur de a en dehors de cette fonction.

4.4.2 Exemple de variables locales

Dans l'exercice 431 précédent, la variable a n'a de valeur que dans la fonction calc.
A la fin du programme principal demander l'affichage du contenu de la variable a et conclure.

4.4.3 Variable globale

Une variable a est dite globale à un script, si elle conserve sa valeur dans la totalité des parties de ce script.
Il faut alors débuter chaque partie du script où la variable a est employée par l'instruction " global a " .

4.4.4 Exemple de variables globales

Modifier les fonctions de l'exercice 431 afin de pouvoir afficher le contenu de la variable a à la fin du programme principal.

CHAPITRE : 5

STRUCTURES ITERATIVES EN PYTHON sous Pyzo

5.1 / La structure POUR

5.1.1 Exemple

```
#exo511
i=0
u=-24
for i in range(1,11) :
  print(i)
  u=2*u-3
  print(u)
  print('——-')
```

Explications :

Ce programme détermine les 10 premiers termes de la suite (u_n) définie par $u_0 = -24$ et $\forall\ n \in \mathbb{N}$, $u_{n+1} = 2.u_n - 3$
La variable i prend les valeurs succéssives de 1 à 10.
Les actions qui sont dans la boucles for sont effectuées 10 fois.
Remarque : vérifier que la dernière valeur prise par i est 10
for j in range (7,4,-1) fournit pour valeurs de j : 7, 6 puis 5 (par pas de -1)

Exercice512

Ecrire une fonction exo512 de paramètre n qui entre la valeur de u_0 puis qui retourne la valeur du $n^{ième}$ terme u_n de la suite (u_n) précédemment définie.

Exercice513

Ecrire une fonction exo513 de paramètre n qui calcule puis affiche les n premiers termes de la suite de Héron définie par $u_0 = \frac{3}{2}$ et
$\forall\ n \in N$, $u_{n+1} = \frac{1}{2}(\ u_n + \frac{2}{u_n})$
Que constate-on ?

.

Evolution 514 : *Programmer* $u_0 = x$ *et* $\forall\ n \in N$, $u_{n+1} = \frac{1}{2}(\ u_n + \frac{x}{u_n})$, *tester !*

Evolution 515 : *Déterminer le nombre d'itérations nécéssaires afin d'obtenir une approximation de* $\sqrt{2}$ *avec* 12 *décimales exactes.*

Evolution 516 : *Démontrer l'exactitude des constatations 513*

Soit (u_n) la suite définie par : $u_0 = \frac{3}{2}$ et $\forall\ n\ \in\ \mathbb{N}$, $u_{n+1} = \frac{1}{2}(u_n + \frac{2}{u_n})$

1 Déterminer $u_1$et u_2.
2 Montrer que $\forall\ n\ \in\ \mathbb{N}$, $u_n > 0$.
3 Rechercher l'éventuelle limite de la suite (u_n).
4 Montrer que $\forall\ n\ \in\ \mathbb{N}$, $\sqrt{2} < u_n$.
5 Etudier le sens de variation de la suite (u_n).
6 Soit (v_n) la suite définie par : $\forall\ n\ \in\ \mathbb{N}$, $v_n = \frac{u_n - \sqrt{2}}{u_n + \sqrt{2}}$.
Montrer : $\forall\ n\ \in\ \mathbb{N}\ \ v_n = (v_0)^{2^n}$.
7 Montrer : $\frac{7}{5} < \sqrt{2}$.
8 Montrer : $v_0 < \frac{1}{29}$.
9 Déterminer le plus petit entier n tel que $u_n < 10^{-14} + \sqrt{2}$.

Exercice517

Ecrire une fonction exo517 de paramètre n qui calcule puis affiche la somme des *n premiers entiers naturels.*

Exercice518

Ecrire une fonction exo518 de paramètres a, b et n qui calcule puis affiche les n premiers termes de la suite de Fibonacci définie par : $u_1 = a$, $u_2 = b$
$\forall\ n\ \geq 1$, $u_{n+2} = u_{n+1} + u_n$
Remarque : utiliser la structure $print(u, end = \text{" "})$ afin d'afficher les valeurs successives de u sur une même ligne et séparées par un espace.

.

5.2 / La structure TANT QUE

5.2.1 Exemple

```
#exo521 détermination du rang du premier terme inférieur à -1000
i=0
u=-24
while u>-1000:
    i,u=i+1,2*u-3
print(i,u)
```

Explications :

Ce programme détermine les premiers termes de la suite (u_n) définie par $u_0 = -24$ et $\forall\ n \in \mathbb{N}$, $u_{n+1} = 2.u_n - 3$
Les actions qui sont dans la boucles *while* sont effectuées tant que u_n est supérieur à -1000
Contrairement à une boucle for, on ne sait pas à priori combien de fois les actions de la boucle seront effectuées.
Remarque : l'instruction $i, u = i + 1, 2 * u - 3$ affecte simultanément i et u.
Attention : les actions $\begin{cases} i = i + 1 \\ u = u + i \end{cases}$ et $i, u = i + 1, u + i$ ne fournissent pas les mêmes résultats. Dans le premier cas, la nouvelle valeur de u est la somme de l'ancien u avec le nouveau i calculé à la ligne précédente, alors que dans le second cas, la nouvelle valeur de u est la somme de l'ancien u avec l'ancien i.

Exercice522

Ecrire une fonction exo522 qui retourne le rang du premier terme de la suite (u_n) *précédente tel que* $\frac{u_{n+1}}{u_n} < 2 + 10^{-4}$

Exercice523

Déterminer $N \in \mathbb{N}$ / $\forall n > N, \sum_{i=0}^{n} i^2 > 10^2$

Exercice524

Idem exo 523 mais en utilisant $S_2 = \frac{n(n+1)(2n+1)}{6}$

.

Exercice525

Ecrire une fonction exo525 qui détermine, par dichotomie, une approximation de $\sqrt{2}$
Inclure un compteur afin de déterminer le nombre d'itérations nécéssaires pour obtenir une approximation de $\sqrt{2}$ *avec* 12 *décimales exactes.*
Déterminer par le calcul ce nombre d'itérations, et comparer avec l'exercice 513.

Exercice526

Ecrire l'algorithme de la méthode de résolution d'équation dite
" méthode des tangentes ou de Newton "
Le programmer
Utiliser ce programme pour obtenir une approximation de $\sqrt{2}$ *avec* 12 *décimales exactes et comparer le nombre d'itérations nécéssaires avec les résultats des méthodes 513 et 525.*

CHAPITRE : 6

VECTEURS EN PYTHON sous Pyzo

6.1 Exemples de créations de vecteurs

Un vecteur est une variable comportant plusieurs coordonnées

6.1.1 Exemple

```
# exo611 exemple de création de tableau
A=5*[None]
for i in range(0, 5)
   A[i]=i**2
print(A)
print(i)
```

Explications :

L'instruction A=5*[None] créée un tableau de 5 cases vides (numérotées de 0 à 4)
Ce programme crée un vecteur un vecteur A de 5 coordonnées, $A(4) = 6$

6.1.2 Exemple

```
#exo612 exemple de création de tableau
A=[5,4,3,8,7]
print(A)
print(A[2])
B = A + A
print(B)
print(len(B))
C = A[1 : 4]
print(C)
```

Explications :

A est le vecteur de 5 coordonnées (5 4 3 8 7)
$A(2) = 3$
B est le vecteur de 10 coordonnées (5 4 3 8 7 5 4 3 8 7)
len(B) indique le nombre de coordonnées de B donc ici len(B)=10
C contient le sous vecteur de A défini par C=(4 3 8)

.

Exercice613

Soit A un vecteur contenant des nombres entiers
Ecrire un algorithme qui détermine la valeur de l'élément maximum de A.
Compléter cet algorithme par la recherche de l'indice de ce maximum.
Traduire cet algorithme en Python et le sauvegarder sous le nom exo613

Exercice614

Soit A un vecteur contenant des nombres entiers. Ecrire un algorithme qui range en ordre décroissant les éléménts du tableau A dans un tableau B.

Exercice615

Reprendre l'exercice 614 sans utiliser de tableau annexe B.

Exercice616

Soit T un tableau contenant des nombres entiers. Soit n un nombre saisi au clavier.
Ecrire un algorithme qui détermine si n est présent dans le tableau T.
(éventuellement donner la position de n)
Traduire cet algorithme en Python .

Exercice617

Reprendre l'exercice 616 avec un tableau déjà trié en ordre croissant.

Exercice618

Soit T un tableau contenant des nombres entiers en ordre croissant.
Soit n un nombre saisi au clavier.
Ecrire un algorithme qui insère n dans le tableau T afin que *T*
demeure ordonné croissant.

Exercice619

Reprendre l'exercice 618 à l'aide de fonctions

.

6.3 Exemple de vecteurs

6.3.1 Exemple

```
# exo631 création d'un vecteur à n
# composantes égalemment réparties
import numpy as np
C = np.linspace(2,10,3)
print(C)
```

Explications :

le module *numpy* est le module de calcul scientifique de *Python*.
Il est indispensable pour les travaux matriciels.
Pour plus de rapidité d'écriture *numpy* est nommé *np*
$np.linspace(a, b, n)$ crée un vecteur à n coordonnées également réparties,
la première des coordonnées étant a et la dernière étant b.
L'appel de la fonction $np.linspace(2, 10, 5)$ affiche : 2 4 6 8 10 soit 5 valeurs également réparties entre 2 et 10.
L'appel de la fonction $np.linspace(2, 10, 4)$ affiche : 2 $4,66$ $7,33$ 10 soit 4 valeurs également réparties entre 2 et 10.

6.3.2 Exemple de tracé de fonction

```
#exo632
from pylab import *          importation du module de tracé de courbe
y=input(' Entrer votre fonction ')          saisie de la fonction
f=lambda x:eval(y)          création de la fonction f
A = arange(20)          A est un tableau de 20 nombres entiers, de 0 à 19
plot(A,f(A))          représentation de y = f(x) pour x ∈ [0;19]
show()          visualisation de la représentation de la fonction.
```

6.4 Calcul d'aires

Exercice641

Soit f une fonction continue sur $[a, b]$, on pose $I = \int_a^b f(x)dx$.
Ecrire l'algorithme de détermination d'une valeur approchée de I par la méthode dite " méthode des rectangles ou de Riemann ".
Programmer cette méthode en Python.

CHAPITRE : 7

MATRICES EN PYTHON sous Pyzo

7.1 Exemples de créations de matrices

Une matrice est une variable comportant plusieurs dimensions et pour chacune de ses dimensions, plusieurs coordonnées. Pour effectuer du calcul matriciel il faut débuter le programme par l'instruction : *import numpy as np*

7.1.1 Exemples

A =np.array([[1,2],[3,4]]) donne la matrice $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$

B= np.ones((3,3)) donne $B = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$

C=np.zeros((2,3)) donne $C = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$

D=np.eye(3) donne $D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

E=np.random.rand(2,3) crée une matrice à 2 lignes et 3 colonnes dont les coefficients sont des nombres aléatoires de $[0;1[$.

Exercice712

Ecrire un algorithme qui détermine dans une matrice carrée C les premiers coefficients du triangle de pascal.

7.2 Opérateurs matriciels prédéfinis

7.2.1 La somme terme à terme

Soient $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ et $B = \begin{pmatrix} 7 & 8 & 9 \\ 1 & 2 & 3 \end{pmatrix}$ alors $A + B = \begin{pmatrix} 8 & 10 & 12 \\ 5 & 7 & 9 \end{pmatrix}$

7.2.2 Le produit matriciel au sens de l'algèbre linéaire : *np.dot(A,B)*

Soient $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ et $B = \begin{pmatrix} 4 & 5 & 6 & 7 \\ 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \end{pmatrix}$ alors np.dot(A,B)$= \begin{pmatrix} 12 & 18 & 24 & 30 \\ 33 & 48 & 63 & 78 \end{pmatrix}$

7.2.3 La transposée de A $A.T$

Soit $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ $A.T = \begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix}$

7.2.4 Inverse d'une matrice *np.linalg.inv(A)*

Soit $A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$ np.linalg.inv(A)= $\begin{pmatrix} -2 & 1 \\ 1.5 & -0.5 \end{pmatrix}$

7.2.5 Dimensions des matrices *A.shape*

Soit $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ (m,n)=A.shape fournit $m = 2$ et $n = 3$

7.3 Exercices matriciels classiques

Exercice731

Ecrire un algorithme qui détermine, lorsque cela est possible, la somme de deux matrices. Le programmer sans utiliser d'instruction prédéfinie !

Exercice732

Ecrire un algorithme qui détermine, lorsque cela est possible, le produit matriciel de deux matrices. Le programmer sans utiliser d'instruction prédéfinie !

Exercice733

Ecrire l'algorithme de la concaténation verticale de deux matrices.

Exercice734

Simuler une loi uniforme (lancer de dé)

Exercice735

Simuler une loi binômiale (10 lancers d'une pièce de monnaie)

Exercice736

Programmer les formules d'une loi binômiale (10 lancers d'une pièce de monnaie)

CHAPITRE : 8

TRAITEMENT D'IMAGE

EN PYTHON sous Pyzo

Les images numériques en noir et blanc peuvent être considérées comme des matrices à deux dimensions (largeur et hauteur). Chaque élément de cette matrice est la luminosité d'un pixel de l'image (0 pour noir et 1 pour blanc)

De même, les images numériques en nuances de gris peuvent être considérées comme des matrices à deux dimensions. Chaque élément de cette matrice est la luminosité d'un pixel de l'image (de 0 pour noir à 255 pour blanc)

Les images numériques en couleurs peuvent être considérées comme la superposition de trois couches de matrices à deux dimensions. La première matrice code la luminosité du Rouge, la seconde celle du Vert et la troisème celle du Bleu.

Attention les pixels commencent à (0,0) en haut à gauche et sont orientés : $\overset{x}{\longrightarrow}$ $y\ \downarrow$

8.1 Exemples de travaux sur une image en noir et blanc

8.1.1 Exemple 1

Taper le code du programme exo811 ci-dessous et en demander l'exécution.
Attention : Ne pas taper les commentaires et modifier le chemin d'acces au fichier A1.jpg

```
# exo811    1ére gestion de la lettre A
from PIL import Image
fichier1 = "G:/chap8/A1.JPG"   # décrire l'emplacement de l'image
img1 = Image.open(fichier1)  # pour charger A1.jpg
img1.show( )   # pour afficher l'image
(L,H)=img1.size   # pour obtenir la largeur L, et la hauteur H, de l'image
print(img1.format)   # pour afficher le format de l'image : jpg, bmp, pgm, gif ...
print (L,H)
data=list(img1.getdata())   # pour obtenir la liste des pixels
print(len(data))   # pour imprimer le nb de pixels de l'image
print(data[α:β])   # pour imprimer les pixels de α à β
img1=img1.convert('L')   # pour convertir img1 en nuances de gris
img1.save('G:/chap8/A1.pgm')  # pour sauvegarder l'image au format pgm
img2 = Image.open('G:/chap8/A1.pgm')   # pour charger A2.pgm
(L,H)=img2.size
print(img2.format)
print (L,H)
data=list(img2.getdata())
print(len(data))
print(data[α:β])
```

Exercice812 Ajout d'un rectangle noir

Charger en mémoire l'image A1.pgm.
A l'aide de l'instruction $img1.putpixel((i,j),0)$ et de deux boucles imbriquées, modifier cette image afin de lui adjoindre un rectangle noir de 20 pixels de large et de 40 de haut, situé en haut à gauche du A. Visualiser le résultat.

Exercice813 Création d'une image négative

Charger en mémoire l'image A1.pgm.
Modifier cette image afin de permutter les pixels noirs et blancs.
Visualiser le résultat.

8.2 Exemples de travaux sur une image en nuances de gris

Exercice821 transformation d'une image couleur en nuances de gris.

Charger en mémoire un fichier image couleur nommé image1 jpg.
Visualiser cette image couleur et en afficher ses dimensions et nombre de pixels.
Afficher 30 pixels consécutifs et représentatifs des couleurs de cette image.
A l'aide de l'instruction $img2 = img1.convert('L')$, convertir l'image couleur en image "nuances de gris".
Visualiser le résulat.
A l'aide de l'instruction $img2.save('$chemin /image2.pgm') sauvegarder l'image grise sous le nom image2.pgm

Charger en mémoire le fichier image grise nommé image2.pgm.
Visualiser cette image grise et en afficher ses dimensions et nombre de pixels.
Afficher les mêmes 30 pixels consécutifs et représentatifs des nuances de cette image.
Comparer avec les résultats précédents.

Exercice822 Eclaircissement d'une image en nuances de gris

Charger en mémoire le fichier image grise nommé image2.pgm.
Demander la valeur n de l'éclaircissement (de 1 à 255).
L'instruction $c1 = img1.getpixel((i,j))$ permet de retourner la valeur de l'intensité lumineuse du pixel de coordonnées (i,j) dans la variable $c1$.
A l'aide de deux boucles imbriquées, modifier cette image afin de baisser, de la valeur n choisie, l'intensité lumineuse de chaque pixel.
L'instruction $img1.putpixel((i,j),c2)$ modifie l'intensité du pixel (i,j) de la valeur $c1$ à la valeur $c2$.
L'utilistation de la fonction $\min(a,b)$ est nécessaire afin de ne pas dépasser 255.
Visualiser le résultat.

Exercice823 Augmentation du contraste d'une image en nuances de gris

Charger en mémoire le fichier image grise nommé image2.pgm.
Demander la valeur p du coefficient de contraste (par exemple $p = 2$).
Pour chaque pixel, multiplier par p la valeur de l'écart entre sa luminosité et la moyenne 127.
L'utilistation des fonctions $\min(a,b)$ ou $\max(a,b)$ est nécessaire afin de ne dépasser ni 0 ni 255.
Visualiser le résultat.

Exercice824 seuillage d'une image en nuances de gris

Charger en mémoire le fichier image grise nommé image2.pgm.
Demander la valeur s du seuil de visualisation des pixels.
Modifier l'intensité lumineuse de chaque pixel afin qu'au delà de s, les pixels soient affichés en blanc.
Visualiser le résultat.

Exercice825 mise sur fond gris gris

Charger en mémoire le fichier A1.pgm.
Modifier l'intensité lumineuse de chaque pixel afin qu'au delà d'une intensité de 200, les pixels soient affichés en gris intensité 130.
Visualiser le résultat.
Sauvegarder la nouvelle image sous le nom A2.pgm.

Exercice826 Détection des bordures d'une image en nuance de gris

Charger en mémoire le fichier A2.pgm.
Ecrire un programme qui met dans une nouvelle image A3 un pixel à 255 si l'écart-type des 4 points qui entourent le pixel (haut, bas, droit et gauche) de l'image d'origine est supérieur à un seuil (par exemple 10).
identique à A2.
Sauvegarder en A3.pgm
Visualiser le résultat.

Exercice827 Floutage d'une image en nuance de gris

Charger en mémoire le fichier A2.pgm.
Pour chaque pixel (i,j) de A2, déterminer la moyenne des $(2c+1)^2$ pixels formant un domaine carré 'centré sur (i,j)'. Faire des essais pour $c \in \{1, 2, 3\}$
Créer une image qui contient, à la place du pixel(i,j), la moyenne de ces $(2c+1)^2$ pixels.
Sauvegarder en A4.pgm
Visualiser le résultat.

Exercice828 Agrandissement d'une image en nuance de gris

Charger en mémoire le fichier A2.pgm.
Remplacer chaque pixel par un carré de côté 2
Sauvegarder en A5.pgm
Visualiser le résultat.

Exercice829 Symétrique d'une image en nuance de gris

Charger en mémoire le fichier A2.pgm.
En considérant l'axe vertical au mileiu de l'image, créer à droite, le symétrique de l'image de gauche.
Sauvegarder en A6.pgm
Visualiser le résultat.