

L'objectif principal de ce second TP est d'utiliser la structure « *if .. elif ...else* : » vue en cours dans des situations différentes. On aborde aussi la notion de fonctions.

On demande de rédiger un compte-rendu au format *.doc* ou *.odt* à transférer en fin d'activité par l'intermédiaire de l'onglet **Mon Compte** du site <https://nsibrantly.fr> en utilisant le code : **tp3** . Ce compte-rendu contiendra :

- les réponses aux différentes questions posées,
- les captures d'écran **des morceaux de codes** écrits et celles **des résultats des exécutions**. Pour faire ces captures, utiliser l'*Outil Capture d'écran* de Windows (touches clavier *windows+Shift+s*)

Attention à repérer correctement les titres de paragraphe.



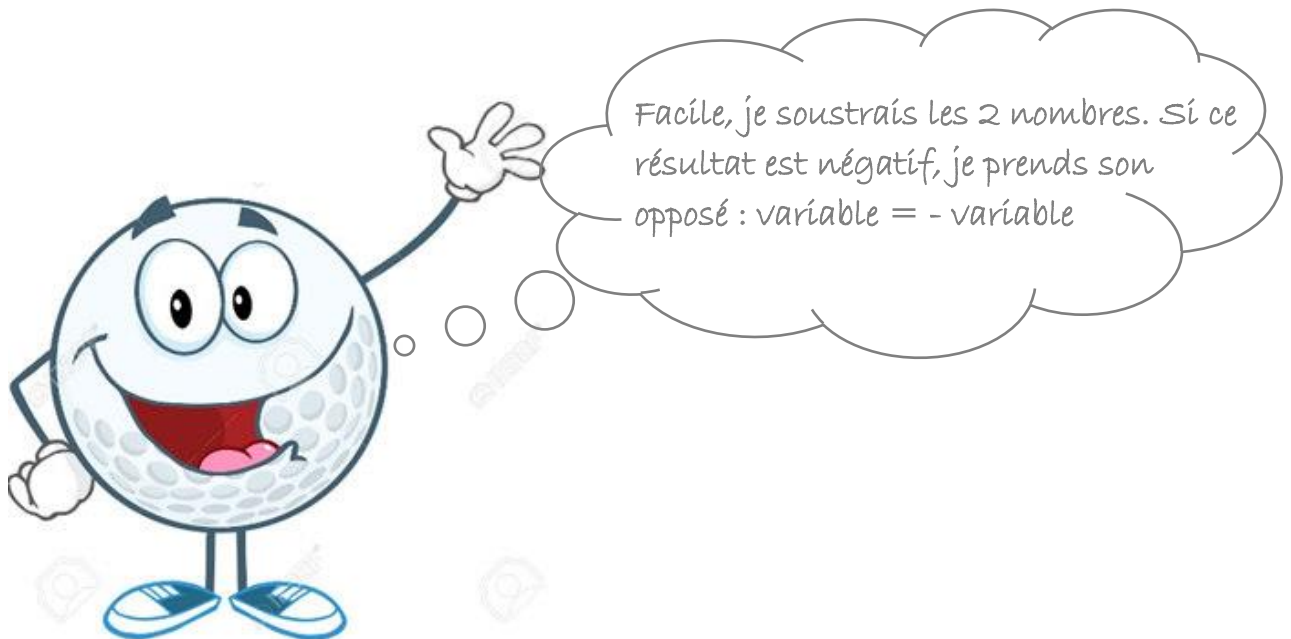
## 1- ECART ENTRE 2 NOMBRES :

⇒ Ecrire un programme qui demande 2 nombres à l'utilisateur et affiche en retour l'écart entre ces nombres (différence sans signe). Attention, le premier nombre donné ne sera pas nécessairement le plus petit des deux.

Exemples d'exécution avec comme valeurs saisies 5 et 1 et ensuite 1 et 5 : →

```
>>> (executing file "ifElifElse.py")
Premier nombre : 5
Second nombre : 1
Ecart : 4.0

>>> (executing file "ifElifElse.py")
Premier nombre : 1
Second nombre : 5
Ecart : 4.0
```



## 2- FONCTION QUI RENVOIE LA VALEUR ABSOLUE D'UN NOMBRE :

⇒ Ecrire le code ci-contre dans lequel on définit une fonction nommée pasDeSigne().

⇒ Exécuter ce code (touche F5) afin que Pyzo lise et mémorise cette nouvelle fonction que vous avez créée.

⇒ Dans la console, exécuter les lignes

```
>>> pasDeSigne(20)
>>> pasDeSigne(-20)
>>> pasDeSigne(1-5)
```

```
def pasDeSigne(nombre) :
    if nombre < 0 :
        nombre = - nombre
    return nombre
```

Bien respecter les indentations, le : et les mots clé python def et return

... on voit ainsi qu'en exécutant pasDeSigne(..), le nombre mis entre les parenthèses est retourné sans signe. Ainsi par exemple pasDeSigne(-20) prend comme valeur 20 .

⇒ Dans la console toujours exécuter les lignes :

```
>>> age = pasDeSigne(2007-2023)
>>> age
16
```

... on voit ici qu'une variable age est créée. La valeur de pasDeSigne(2007-2023) qui est de 16 est mémorisée dans cette variable age.

## 3- FONCTION PRECEDENTE APPELEE PAR UN PROGRAMME PRINCIPAL :

⇒ Exécuter le code ci-contre constitué de 2 parties :

Partie où sont écrites la ou les fonctions. Python commence sa lecture du fichier par le haut, donc par cette partie. Les scripts des fonctions sont lus en premier et sont mémorisés SANS ETRE EXECUTES.

Partie appelée « programme principal ». Les fonctions définies précédemment et déjà mémorisées sont ici exécutées au milieu d'autres commandes.

```
# fonctions
def pasDeSigne(nombre) :
    if nombre < 0 :
        nombre = - nombre
    return nombre

# programme principal
a = float(input("Premier nombre : "))
b = float(input("Second nombre : "))
ecart = pasDeSigne(a-b)
print(f"Ecart : {ecart}")
```

## 4- UN PEU DE COULEURS DANS LA CONSOLE : PAS BESOIN DE METTRE DE COPIE D'ECRAN POUR CE PARAGRAPHE

Ecrire avec des couleurs dans la console n'est pas d'une grande utilité. Python n'a pas prévu de procédures simples pour le faire. Cela peut par contre s'avérer utile. On montre ici comment y arriver tout de même, en **préfixant** la chaîne de caractère à afficher, d'un code Ansi : \033[0;31m pour afficher en rouge, \033[0;34m pour afficher en bleu, \033[0;32m pour afficher en vert. On voit cela ci-dessous :

⇒ Dans la console, exécuter :

```
>>> print("NSI")
>>> print("\033[0;31mNSI")
```

Copié\_collé :

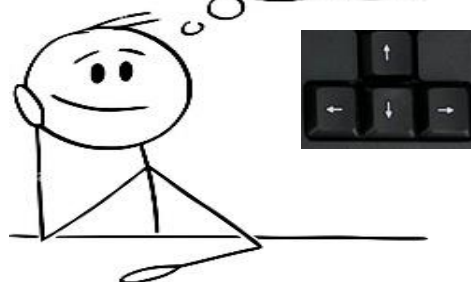


```
print("\033[0;31mNSI")
```

On continue à voir découvrir cela en exécutant dans la console les lignes qui suivent :

```
>>> print("\033[0;34mNSI")
>>> print("\033[0;32mNSI")
```

Utilise les touches haut et bas pour retrouver l'historique des commandes :



```
>>> print("\033[0;31mN \033[0;32mS \033[0;34mI")
```

Copié collé :



```
print("\033[0;31mN \033[0;32mS \033[0;34mI")
```

```
>>> r = "\033[0;31m"
>>> g = "\033[0;32m"
>>> b = "\033[0;34m"
>>> print(r+"N"+g+"S"+b+"I")
```

On continue par la même exécution en écriture formatée :

```
>>> print(f"{r}N{g}S{b}I")
```

Pour revenir à la couleur par défaut de Pyzo, on utilise le préfixe : `\033[0;30mN`

#### 5- ENTREES-SORTIES DANS UNE FONCTION :

Le code ci-contre est mal indenté.

⇒ Copier-coller ce code dans l'éditeur de pyzo et enregistrer le fichier.

⇒ Modifier et compléter le code afin qu'il donne à

l'exécution :

```
>>> (executing file "ifElifElse.py")
bougre de zouave
u

coucou
bachibouzouk
o

coucou
```

```
def estPresent(phrase , lettre) :
print(phrase)
print(lettre)
print()
return 'coucou'

print(estPresent("bougre de zouave","u"))
print(estPresent("bachibouzouk","o"))
```




## 6- FONCTION QUI RECONNAIT UN CARACTERE DANS UN STRING :

Le code ci-contre est mal indenté et comporte 1 erreur.

⇒ Copier-coller ce code dans l'éditeur de pyzo et enregistrer le fichier.

⇒ Modifier et compléter le code afin qu'il donne à l'exécution :

```
>>> (executing file "ifElifElse.py")
bougre de zouave
bachibouzouk
```



```
def estPresent(phrase , lettre) :
rouge = "\033[0;31m"
noir = "\033[0;30m"
new = ""
for c in phrase :
if c == lettre :
new = new + rouge + c
else :
new = new + c
return new


# programme principal
print(estPresent("bougre de zouave","u"))
print(estPresent("bachibouzouk","o"))
```

## 7- FONCTION QUI COMPTE LE NOMBRE D'OCCURRENCE D'UN CARACTERE DANS UN STRING :

Le code ci-contre est mal indenté et comporte 3 erreurs.

⇒ Copier-coller ce code dans l'éditeur de pyzo et enregistrer le fichier.

⇒ Modifier et compléter le code afin qu'il donne à l'exécution :



```
# fonctions
nombreFois(phrase , lettre) :
rouge = "\033[0;31m"
noir = "\033[0;30m"
new = ""

for c in phrase :
if c == lettre
new = new + rouge + c
N = N + 1
else :
new = new + noir + "_"
print(new, end="\033[0;30m : ")
return N

# programme principal
print(nombreFois("bachibouzouk","o"),"fois")
print(nombreFois("Bougre d'extrait de cornichon","e"),"fois")
```

```
>>> (executing file "ifElifElse.py")
  o  o  : 2 fois
  e  e  e  : 3 fois
```



## 8- LECTURE DANS UN FICHER AU FORMAT .TXT :

Le code écrit précédemment s'avère bien plus efficace lorsqu'il est appliqué à une phrase très longue. On se propose ici de créer une phrase qui contient la totalité de la déclaration universelle des droits de l'homme élaborée le 26 août 1789.

⇒ Télécharger le zip donné sur [nsibrantly.fr](http://nsibrantly.fr), le **décompresser** dans le répertoire dans lequel vous travaillez, c'est-à-dire celui dans lequel se trouve les codes .py que vous êtes en train d'écrire.

⇒ Ecrire et exécuter le code ci-contre :

⇒ Dans la console, écrire `>>> phrase`

Afin d'afficher la valeur de la variable  
« *phrase* »

```
# fonctions
def lectureFichier(fichier) :
    f = open(fichier,"r",encoding='utf-8')
    toutLeFichierEnString = f.read()
    f.close()
    return toutLeFichierEnString

# programme principal
phrase = lectureFichier("declaration.txt")
```



Trop facile de lire un fichier :

- On ouvre le fichier
- On place son contenu dans un string avec la méthode `read()`
- On ferme le fichier

## 9- COMBIEN DE MOTS DANS LA DECLARATION UNIVERSELLE DES DROITS DE L'HOMME ?

⇒ Le code ci-contre est mal indenté et comporte 3 erreurs.

⇒ Copier-coller ce code dans l'éditeur de pyzo et enregistrer le fichier.

⇒ Modifier et compléter le code afin qu'il donne à l'exécution :

```
a_a
a_a
a_a
a_a
a_a
a_a
a_a
a_a
a_a
a_a
: 216 fois
```

# fonctions

```
def lectureFichier(fichier) :
f = open(fichier,"r",encoding='utf-8')
toutLeFichierEnString = f.read()
f.close()
return
```

```
def nombreFois(fichier , lettre) :
= lectureFichier(fichier)
```

```
rouge = "\033[0;31m"
```

```
noir = "\033[0;30m"
```

```
new = ""
```

```
N = 0
```

```
for c in phrase :
```

```
if c == lettre :
```

```
new = new + rouge + c
```

```
N = N + 1
```

```
else :
```

```
new = new + noir + "_"
```

```
print(new, end="\033[0;30m : ")
```

```
# programme principal
```

```
print(nombreFois("declaration.txt","a"),"fois")
```



⇒ Changer 1 lettre dans ce code, sur la dernière ligne, pour répondre à la question : « *Combien de mots dans la déclaration universelle des droits de l'homme* » ?

## 10- CHANGER LES LETTRES D'UN TEXTE :

⇒ Le code donné ci-après est mal indenté et est incomplet pour le script de la fonction modifTexte()

⇒ Copier-coller ce code dans l'éditeur de pyzo et enregistrer le fichier.

⇒ Modifier et compléter le code afin

qu'il donne à l'exécution :

```
>>> (executing file "ifElifElse.py")
eyjoyrd'hyo c'est lyndo 2 octobre 2023
```



```
## Fonctions
def couleurs() :
rouge = "\033[0;31m"
vert = "\033[0;32m"
bleu = "\033[0;34m"
noir = "\033[0;30m"
return rouge,vert,bleu,noir

def modifTexte(phrase) :
rouge,vert,bleu,noir = couleurs()
phrase_modifiee = ""
for lettre in phrase :
if lettre == "a" :

# programme principal
new = modifTexte("aujourd'hui c'est lundi 2 octobre 2023")
print(new)
```

#### 11- METTRE LA DECLARATION UNIVERSELLE DES DROITS DE L'HOMME EN COULEUR :

⇒ Afficher dans la console la déclaration universelle des droits de l'homme élaborée le 26 août 1789 :

- avec toutes les voyelles en rouge,
- les lettres "m" et "n" en bleu,
- les autres caractères en noir.