TD 1

Ibrahim ALAME

26/09/2023

1 Expression booléenne

1. Écrire un programme `sort.py` qui prend les valeurs de trois variables a, b et c et les affiche dans l'ordre croissant.

```
1
2  # 2.  Tri simple
3
4  a = ........
5  b = ........
6  c = ........
7  if a > b:
8        ........
9  if a > c:
10       ........
11 if b > c:
12       ........
13 print(a, b, c)
```

Exemple d'exécution :

```
> a=2
> b=7
> c=4

2 4 7
```

2. Écrire un programme `leap.py` qui prend un argument de ligne de commande entier positif représentant une année et affiche `True` si l'année est bissextile et `False` sinon. Une année bissextile est une année divisible par 4 mais pas par 100, ou est divisible par 400. Ne pas utiliser une instruction conditionnelle `if` dans votre programme. Vous auriez besoin d'écrire une expression booléenne appropriée qui évalue la bonne réponse vrai / faux.

```
1  # 3. Année bissextile
2  a = ........
3  B = ........
4  print(B)
```

Exemples :

```
>  1980
True
>  1900
False
```

2 Opérateurs arithmétique

1. Écrire une fonction qui prend un entier en paramètre et qui l'affiche à l'envers. Par exemple, si on l'appelle avec 123456, la fonction affiche 654321. Pour cela il faudra utiliser la division et le modulo. Rappel : 153 % 10 = 3 et 153//10 = 15.

```
# 4. Opérateur arithmétique

import sys
a = ........
while  ........ :
    print( ........ ,end='')
    a= ........
print()
```

2. Écrire un programme `day.py` qui accepte une date comme entrée et écrit le jour de la semaine de la date donnée. Votre programme doit accepter trois arguments de ligne de commande : m (mois), d (jour) et a (année). Pour m, utilisez 1 pour janvier, 2 pour février, et ainsi de suite. Pour la sortie, écrivez 0 pour dimanche, 1 pour Lundi, 2 pour mardi, et ainsi de suite. Utilisez les formules suivantes, pour le calendrier grégorien, et notez que toutes les divisions sont censées être des nombres entiers (l'opérateur //).

$$\begin{aligned} a_0 &= a - (14 - m)/12 \\ x &= a_0 + a_0/4 - a_0/100 + a_0/400 \\ m_0 &= m + 12 \times ((14 - m)/12) - 2 \\ d_0 &= (d + x + (31 \times m_0)/12) \mod 7 \end{aligned}$$

```
# 5. Jour de la semaine

L =("Dimanche", "Lundi", "Mardi", "Mercredi", "Jeudi", "Vendredi", "Samedi")
d = ........
m = ........
y = ........
y0 = ........
x =  ........
m0 = ........
d0 =  ........

print(L[d0])
```

Exemple :

```
d=12
m=7
y=1963
Vendredi
```

3. Écrire un programme `qroots.py` qui lit trois réels a, b, c depuis la ligne de commande puis il calcule et affiche les racines d'un polynôme du second degré $ax^2+bx+c=0$. Indication : `math.sqrt()` est la fonction intégrée qui renvoie la racine carrée d'un nombre.

```
# 6. Equation de second degré
import math as m

a =  .............
b =  .............
c =  .............
d =  .............
if d > 0:
    r1 = (-b-math.sqrt(d))/(2*a)
    r2 =  (-b+math.sqrt(d))/2/a
    print('Deux racines:', r1, ' et ', r2)
else:
    if d == 0:
        print('Une racine double:',  -b/2*a )
    else:
        z =  (-b+math.sqrt(-d)*1j)/2/a
        print('Deux racines complexes:', z, ' et ', z.conjugate())
```

3 Chaîne de caractères

1. Écrivez un programme qui prend un caractère comme argument et affiche s'il s'agit d'une voyelle ou non. Votre programme ne peut contenir que des affectations et des expressions booléennes, sans instructions conditionnelles.

```
# 7. voyelle
n = ..............
print(n ..... "aeiouy")
```

Exemples :

```
> n = e
True
>  n = k
False
```

2. Écrivez un programme `voyelles.py` qui prend une chaîne en donnée et affiche le nombre de voyelles contenus dans la chaîne.

```
# 8. Voyelles
n = ..............
def nbVoyelles(mot):
    k=0
    for  .............. :
        if  .............. :
            ..............
    return  .....

print(nbVoyelles(n))
```

3. Écrivez un programme `mostvowels.py` qui lit une liste de chaînes de caractères à partir de la ligne de commande et affiche la chaîne qui contient le plus grand nombre de voyelles, ainsi que son nombre de voyelles. En cas de plusieurs chaînes ayant le même numéro, affichez le premier rencontré.

```
# 9. Voyelles 2
text = ..............
L=text.split()

def nbVoyelles(mot):
    k=0
    for  .............. :
        if  .............. :
            ..............
    return  .....

n=0
m=''
for mot in L:
    x = .............. :
    if  .............. :
        n=..........
        m=..........

print(m,n)
```

```
> text= expressions assignment conditionals iteration drawing
conditionals 5
```

4 Boucles

1. Un palindrome est un mot ou groupe de mots qui peut se lire indifféremment de gauche à droite ou de droite à gauche en gardant le même sens (ex. radar ; laval ; la mariee ira mal ; Roma Amor).. Écrire un programme `palindrome.py` qui lit une chaîne depuis la ligne de commande et détecte s'il s'agit ou non d'un palindrome. Ne pas utiliser une nouvelle chaîne dans votre programme. Vous devez plutôt parcourir la chaîne pour déterminer s'il s'agit d'un palindrome ou non.

   ```
   > python palindrome.py laval
   True
   > python palindrome.py la mariee ira mal
   True
   > python palindrome.py Roma Amor
   True
   ```

2. Écrire un programme `factorial.py` qui lit un entier positif n à partir de la ligne de commande, calcule son factorielle et l'affiche.
3. Écrire un programme `grid.py` qui affiche une grille de nombres $m \times n$. Un exemple de grille 5×6 est illustrée ci-dessous. Indication : la valeur de l'emplacement $(i; j)$ peut être calculée comme $i + mj$.

   ```
   > python grid.py 5 6
   ```

```
0 5 10 15 20 25
1 6 11 16 21 26
2 7 12 17 22 27
3 8 13 18 23 28
4 9 14 19 24 29
```

4. Écrire un programme `triangle.py` qui prend un paramètre de ligne de commande n et affiche un triangulaire d'étoiles rectangle isocèle de coté n similaire à celui illustré ci-dessous.

```
> python triangle.py 6

* * * * * *
. * * * * *
. . * * * *
. . . * * *
. . . . * *
. . . . . *
```

5. Écrivez un programme `arrow.py` qui affiche une forme de flèche de n lignes comme indiqué ci-dessous.

```
> python arrow.py 4

   *
  ***
 *****
*******
```

6. Écrire un programme `prime.py` qui lit un entier positif n à partir de la ligne de commande, calcule un booléen indiquant si n est premier ou non, et affiche un message approprié comme indiqué ci-dessous.

```
> python prime.py 221
221 n'est pas un nombre premier !
```

7. Écrire un programme interactif qui invite l'utilisateur à penser à un entier entre un et mille et demande ensuite à l'utilisateur de répondre honnêtement par vrai ou faux à la question *Est-ce que votre numéro est strictement inférieur à x ?* pour différentes valeurs de x, jusqu'à ce que le programme puisse déduire l'entier choisi.

   Votre programme doit utiliser la stratégie suivante pour deviner : choisissez le milieu de l'intervalle de recherche comme la valeur de x à chaque itération. La fonction Python `input()` peut être utilisée pour obtenir l'entrée de l'utilisateur pendant que le programme est en cours d'exécution.