Algorithmique et Programmation 1 – TD - TP 4
TYPES DE DONNÉES
CORRECTION

Exercice 1 - Exemple simple sur les chaînes de caractères
Quelle est la valeur de `cha` après la suite d'instructions suivantes ?

```
>>> ch = "Avec consonnes"
>>> cha = ch[0]
>>> cha = cha + ch[2]
>>> cha = cha + ch[6]
>>> cha = cha + ch[9]
>>> cha = cha + ch[12]
```

```
>>> cha
'Aeooe'
```

Exercice 2 - Exemple simple sur les chaînes de caractères (2)
Soit le programme suivant :

```
chaine = "Python !"
res = 0

for c in chaine :
    res = res + 1

print(res)
```

1. Ecrire la table de simulation de ce programme

tour de boucle	var. c	var. res
entrée	-	0
tour 1	"P"	1
tour 2	"y"	2
tour 3	"t"	3
tour 4	"h"	4
tour 5	"o"	5
tour 6	"n"	6
tour 7	" "	7
tour 8	"!"	8

2. Que fait ce programme ?

Retourne la longueur de `chaine` (comme la fonction prédéfinie `len`)

3. Modifier ce programme de façon à ce qu'il retourne la chaîne donnée au départ en doublant chaque lettre, soit dans cet exemple `PPyytthhoonn ! !`

```
chaine = "Python !"
res = ""

for c in chaine :
    res = res + c + c

print(res)
```

Exercice 3 - Exemple simple sur les listes

1. Soit la liste `[1, 3, 5, 7, 9, 11, 13, 15, 17]`. Quelle est sa longueur ? A quel indice trouve-t-on la valeur 1 ? La valeur 17 ? La valeur 9 ?

   Longeur : 9. 1 est à l'indice 0, 17 à l'indice 8, 9 à l'indice 4

2. Soit la liste `lst = [2, 2, 3, 3, 4, 5, 7]`. Quelle est sa longueur ? Que vaut l'expression `lst[0]` ? `lst[4]` ? `lst[7]` ?

   Longeur : 7. `lst[0]` = 2 ; `lst[4]` = 4 ; `lst[7]` : erreur, cet indice n'existe pas.

Exercice 4 - Exemple simple sur les listes (2)

Soit le programme suivant :

```
liste = [32, 5, 76, 8, 13, 2, 14, 25]

li = []
lp = []

i = 0

while i < len(liste) :
    if liste[i] % 2 == 0 :
        lp.append(liste[i])
    else :
        li.append(liste[i])
    i = i + 1

print("li : ", li)
print("lp : ", lp)
```

1. Ecrire la table de simulation de ce programme

tour	i	li	lp	condition while	condition if
entrée	0	[]	[]	0 < 8 : `True`	32 % 2 == 0 : `True`
tour 1	1	[]	[32]	1 < 8 : `True`	5 % 2 == 0 : `False`
tour 2	2	[5]	[32]	2 < 8 : `True`	76 % 2 == 0 : `True`
tour 3	3	[5]	[32, 76]	3 < 8 : `True`	8 % 2 == 0 : `True`
tour 4	4	[5]	[32, 76, 8]	4 < 8 : `True`	13 % 2 == 0 : `False`
tour 5	5	[5, 13]	[32, 76, 8]	5 < 8 : `True`	2 % 2 == 0 : `True`
tour 6	6	[5, 13]	[32, 76, 8, 2]	6 < 8 : `True`	14 % 2 == 0 : `True`
tour 7	7	[5, 13]	[32, 76, 8, 2, 14]	7 < 8 : `True`	25 % 2 == 0 : `False`
tour 8	8	[5, 13, 25]	[32, 76, 8, 2, 14]	8 < 8 : `False`	-

2. Que fait ce programme ?

Sépare une liste d'entiers en une liste contenant les entiers pairs, et une liste contenant les entiers impairs

3. Modifier ce programme de façon à ce qu'il analyse un par un tous les éléments d'une liste de chaînes de caractères, et génère une liste contenant les mots de plus de 4 caractères, et une liste contenant les mots de 4 caractères ou moins.

```
liste = ["chat", "chien", "renard", "girafe", "éléphant", "rat", "souris", "serpent"]
grand = []
petit = []
i = 0

while i < len(liste) :
    if len(liste[i]) > 4 :
        grand.append(liste[i])
    else :
        petit.append(liste[i])
    i = i + 1

print("Mots de plus de 4 caractères : ", grand)
print("Mots de 4 caractères ou moins : ", petit)
```

Exercice 4 - Exemple simple sur les ensembles

Donner les valeurs des variables à la fin de la suite d'instructions suivante :

```
set1 = {"a", "b", "c"}
set2 = {1, 2, 3, 4}

set2.update({2, 3, 5, 6})

set3 = set1 | set2

set3.discard(2)
set3.remove("b")

set4 = set1 & set3

set2.clear()

set2 = set3 - set4
```

```
set1 = {'c', 'a', 'b'}
set2 = {1, 3, 4, 5, 6}
set3 = {1, 3, 4, 5, 6, 'c', 'a'}
set4 = {'c', 'a'}
```

Exercice 5 - Liste d'entiers

Ecrire un programme qui, à partir d'une liste d'entiers, calcule puis affiche la somme de tous les entiers de la liste, ainsi que l'élément minimum et l'élément maximum de cette liste

```
liste = [32, 5, 76, 8, 13, 2, 14, 25]

if len(liste) == 0:
    print("La liste est vide")
else:
    min = liste[0]
    max = liste[0]
    somme = liste[0]

    for i in range(1, len(liste)) :
        somme = somme + liste[i]
        if liste[i] < min :
            min = liste[i]
        if liste[i] > max :
            max = liste[i]

    print("La somme des éléments de la liste est", somme, "le max est", max, "et le min est", min)
```

Exercice 6 - Vérification de tri de liste d'entiers

Ecrire un programme qui demande à l'utilisateur de rentrer une liste d'entiers, et qui vérifie si elle est triée ou non. Voici un déroulé du programme attendu :

```
Combien d'entiers comporte votre liste ? 5
Quel est l'entier 0 de votre liste ? 1
Quel est l'entier 1 de votre liste ? 5
Quel est l'entier 2 de votre liste ? 2
Quel est l'entier 3 de votre liste ? 6
Quel est l'entier 4 de votre liste ? 7
Votre liste est la suivante :  [1, 5, 2, 6, 7]
Elle n'est pas triée
```

```
long = int(input("Combien d'entiers comporte votre liste ? "))
liste = []
tri = True

for i in range(long) :
    print("Quel est l'entier", i, "de votre liste ?", end=" ")
    n = int(input())
    liste.append(n)
    if i >= 1 and liste[i] < liste[i-1]:
        tri = False

print("Votre liste est la suivante : ", liste)
if tri :
    print("Elle est triée")
else :
    print("Elle n'est pas triée")
```

Exercice 7 - Inversion d'une chaîne de caractères
Ecrire un programme qui demande une chaîne de caractères à l'utilisateur et qui l'affiche à l'envers. Voici un déroulé du programme attendu :

```
Tapez une phrase : saperlipopette dit le poète
etèop el tid ettepopilrepas
```

```
chaine = input("Tapez une phrase : ")

longueur = len(chaine)
envers = ""

for i in range(len(chaine), 0, -1):
    envers = envers + chaine[i-1]

print(envers)
```

Exercice 8 - Occurences d'un caractère
Ecrire un programme qui, à partir d'une chaîne de caractère et d'un caractère, affiche toutes les positions de la chaîne ou le caractère apparait. Voici un déroulé du programme attendu :

```
Tapez une phrase : saperlipopette dit le poète
De quel caractère souhaitez vous connaitre les occurences ? p
2 7 9 22
```

```
chaine = input("Tapez une phrase : ")
car = input("De quel caractère souhaitez vous connaitre les occurences ? ")

for i in range(len(chaine)) :
    if chaine[i] == car :
        print(i, end=" ")

print(" ")
```

Exercice 9 - Suppression d'un caractère
Ecrire un programme qui demande une chaîne de caractères à l'utilisateur, un caractère particulier, puis supprime toutes les occurences de ce caractère dans la chaîne de caractères. Voici un déroulé du programme attendu :

```
Tapez une phrase : saperlipopette dit le poète
Quel caractère souhaitez vous supprimer ? p
saerlioette dit le oète
```

```
chaine = input("Tapez une phrase : ")
car = input("Quel caractère souhaitez vous supprimer ? ")
res = ""

for c in chaine :
    if c != car :
        res = res + c

print(res)
```

Exercice 10 - Transformation d'une phrase en liste

Ecrire un programme qui demande une chaîne de caractères à l'utilisateur, puis qui place dans une liste les mots de la chaîne (séparés par des espaces). Voici un déroulé du programme attendu :

```
Tapez une phrase : saperlipopette dit le poete
['saperlipopette', 'dit', 'le', 'poete']
```

```
chaine = input("Tapez une phrase : ")
liste = []
mot = ""

for i in range(len(chaine)) :
    if chaine[i] == ' ':
        liste.append(mot)
        mot = ""
    else:
        mot = mot + chaine[i]

liste.append(mot)
print(liste)
```

Exercice 11 - Première et dernière occurences

1. Ecrire un programme qui, à partir d'une liste d'entiers et d'une valeur entière, affiche l'indice de la dernière occurrence de cette valeur dans la liste. Si la liste ne contient pas cette valeur, ce sera clairement affiché.

```
lst = [1,2,3,2,2,4,5,2,3]
n = 8

indice = -1

for i in range(len(lst)) :
    if lst[i] == n :
        indice = i

if indice == -1 :
    print("Le nombre", n, "n'apparait pas dans la liste")
else :
    print("Le nombre", n, "apparait en dernière position à l'indice", indice, "dans la liste")
```

2. Ecrire un programme qui, à partir d'une chaine de caractères et d'un caractère, affiche l'indice de la première occurrence de ce caractère dans la chaine. Si la chaine ne contient pas ce caractère, ce sera clairement affiché.
   Evitez de parcourir le reste de la chaine de caractères une fois le caractère trouvé

```
chaine = input("Tapez une phrase : ")
car = input("De quel caractère souhaitez vous connaitre la première occurence ? ")

i = 0
trouve = False

while i < len(chaine) and not(trouve) :
    if chaine[i] == car :
        print("La première occurence du caractère", car, "est à l'indice", i)
        trouve = True
    i = i+1

if not(trouve):
    print("Le caractère", car, "n'apparait pas dans la chaine")
```

Exercice 12 - Association de listes

Ecrire un programme qui, à partir d'une liste de notes et d'une liste de noms, vérifie que ces deux listes sont de la même taille, et si c'est bien le cas, associe à chaque nom la note qui correspond et la mention obtenue.

Par exemple, si les listes données sont :

`notes = [17, 8, 19, 4, 15, 11, 13]` et

`noms = ["adele", "bernard", "charles", "denise", "eric", "fabio", "ginette"]`,

le résultat affiché doit être :

`[['adele', 17, 'TB'], ['bernard', 8, 'Echec'], ['charles', 19, 'Félicitations'], ['denise', 4, 'Echec'], ['eric', 15, 'B'], ['fabio', 11, 'P'], ['ginette', 13, 'AB']]`

```
notes = [17, 8, 19, 4, 15, 11, 13]
noms = ["adele", "bernard", "charles", "denise", "eric", "fabio", "ginette"]
if not(len(notes) == len(noms)) :
    print("Les liste ne correspondent pas")
else :
    assoc = []
    for i in range(0, len(notes)) :
        if notes[i] < 10 :
            assoc.append([noms[i], notes[i], "Echec"])
        elif notes[i] < 12:
            assoc.append([noms[i], notes[i], "P"])
        elif notes[i] < 14:
            assoc.append([noms[i], notes[i], "AB"])
        elif notes[i] < 16 :
            assoc.append([noms[i], notes[i], "B"])
        elif notes[i] < 18 :
            assoc.append([noms[i], notes[i], "TB    "])
        else :
            assoc.append([noms[i], notes[i], "Félicitations"])
    print(assoc)
```

Exercice 13 - Palindrome

Ecrire un programme qui demande à l'utilisateur de rentrer une chaine de caractères, et qui vérifie si c'est un palindrome, c'est-à-dire une chaîne qui est la même si on la lit de la gauche vers la droite ou de la droite vers la gauche. Voici des déroulés du programme attendu :

```
Tapez une phrase : ballab
C'est un palindrome
Tapez une phrase : balalab
C'est un palindrome
Tapez une phrase : blaalab
Ce n'est pas un palindrome
```

```
chaine = input("Tapez une phrase : ")
long = len(chaine)
i = 0
palindrome = True
while i < long//2 and palindrome :
    if not(chaine[i] == chaine[long-i-1]) :
        palindrome = False
    i = i+1
if palindrome:
    print("C'est un palindrome")
else:
    print("Ce n'est pas un palindrome")
```

Exercice 14 - Nombre d'occurences du maximum

Ecrire un programme qui, à partir d'une liste d'entiers, détermine le nombre maximum de la liste, et le nombre d'occurences de cet élément.

Attention, la liste d'entiers ne devra être parcourue qu'une seule fois.

```
liste = [32, 250, 76, 8, 76, 32, 76, 250]

max = liste[0]
nb_occur = 1

for i in range(1, len(liste)) :
    if liste[i] > max :
        max = liste[i]
        nb_occur = 1
    elif liste[i] == max :
        nb_occur = nb_occur + 1

print("Le max de la liste est", max, "il apparait", nb_occur, "fois dans la liste")
```

Exercice 15 - Répétitions

Ecrire un programme qui, à partir d'une liste, affiche l'ensemble des éléments répétés au moins une fois.

Voici des déroulés du programme attendu :

```
Liste : [32, 25, 76, 8, 76, 32, 76, 25]
Ensemble des éléments répétés : {32, 25, 76}

Liste : [1, 2, 3, 4]
Ensemble des éléments répétés : set()
```

```
#liste = [32, 25, 76, 8, 76, 32, 76, 25]
liste = [1, 2, 3, 4]
s = set()

for i in range(len(liste)) :
    if liste[i] in liste[i+1:] :
        s.add(liste[i])

print("Liste :", liste)
print("Ensemble des éléments répétés :", s)
```