TP - Les chaînes de caractères - En plus

M. Tellene

Exercice 1

Pour cet exercice, il faudra modifier le fichier **deux_chaines_une.py**

Dans la même optique que l'exercice précédent, le programme devra en plus afficher un mot construit avec la moitié des deux autres mots. Le mot final sera donc égal à la première moitié de `mot1` et à la deuxième moitié de `mot2`.

```
mot1 = "haricot"
mot2 = "magique"

#Résultat du programme

haue #deux premières lettres et deux dernières
harique #première moitié et deuxième moitié
```

Aide : renseignez-vous sur le « slicing » des chaînes de caractères

Exercice 2

Pour cet exercice, il faudra créer un fichier **extraction_caractere.py**

Écrire un programme qui initialise deux variables : une chaîne de caractères (`mot` par exemple) et un nombre (`x` par exemple). Le programme doit afficher la lettre étant à l'indice `x` dans `mot`. Pour rappel, les indices commencent à 0 en Python.

Exercice 3

Pour cet exercice, il faudra modifier le fichier **extraction_caractere.py**

En plus d'afficher la lettre étant à l'indice `x` lettre dans `mot`, le programme devra afficher le mot **sans** la lettre à l'indice x. Nous partirons du principe que `x` sera toujours positif.

Exemple de rendu :

```
mot = "bonjour"
x = 2

#Résultat du programme

La lettre à indice 2 dans bonjour est n
Le nouveau mot est : bojour
```

Attention, votre programme devra gérer le cas où l'utilisateur entre une chaîne vide et le cas où l'indice est plus grand que la longueur de la chaîne!

Exercice 4

Pour cet exercice, il faudra créer un fichier **retourner_chaine.py**

Écrire un programme qui initialise une variable avec une chaîne de caractère. Le programme devra ensuite afficher le mot, mais à l'envers.

Exemple de rendu :

```
mot = "bonjour"

#Résultat du programme

Le mot retourné est : ruojnob
```

Pour cet exercice, nous allons avoir besoin d'une boucle `for`. Cette boucle permettra de parcourir le mot. Une chaîne de caractères Python est dite **itérable**, cela veut dire qu'on peut **itérer (parcourir)** chacun des caractères de la chaîne.

1. Nous allons commencer par afficher toutes les lettres d'une chaîne de caractères. Pour ce faire écrire le code suivant :

```
chaine = "salut"
for lettre in chaine:
    print(lettre)
```

2. Écrire en commentaire ce que la console affiche quand on exécute ce programme
3. Modifier votre programme pour créer une variable égale à une chaîne vide. Pour rappel, une chaîne vide Python se crée de la manière suivante : `chaine_vide = ""`
4. Modifier votre programme comme ceci :

```
chaine = "abcdefghijklmnopqrstuvwxyz"
chaine2 = ""
for lettre in chaine:
    chaine2 = lettre

print(chaine2)
```

5. Écrire en commentaire ce que la console affiche à la fin programme. C'est la valeur de `chaine2`.
6. En utilisant la **concaténation** de chaînes, résoudre l'exercice

EXERCICE 5

Pour cet exercice, il faudra créer un fichier ascii.py

Écrire un programme qui initialise une variable avec une chaîne de caractère. Le mot devra ensuite être converti en binaire. Pour ce faire, vous allez convertir d'abord les caractères de la chaîne **en ASCII**.

Exemple de rendu :

```
mot = "salut"
s : 0b1110011
a : 0b1100001
l : 0b1101100
u : 0b1110101
t : 0b1110100
```

Encore une fois, nous allons devoir parcourir la chaîne de caractères (cf. exercice précédent)

1. Écrire une boucle qui parcoure la chaîne et produit le résultat suivant :

```
>>> Entrer un mot : salut
s
a
l
u
t
```

2. Une fois fait, modifier votre programme pour qu'il transforme ces caractères en caractères ASCII. Pour ce faire, vous allez utiliser la fonction `ord(...)`. Cette fonction permet d'**obtenir le code ASCII d'un caractère**. Par exemple `ord("x")` donne le code ASCII du caractère `x`. L'affichage produit devra être celui-ci :

```
>>> Entrer un mot : salut
s : 115
a : 97
l : 108
u : 117
t : 116
```

3. Enfin, en utilisant la fonction `bin(...)`, transformer le code ASCII en binaire. Cette fonction permet d'**obtenir le code binaire d'un entier**. Par exemple `bin(115)` donne le code binaire de 115. L'affichage produit devra être celui-ci :

```
>>> Entrer un mot : salut
s : 0b1110011
a : 0b1100001
l : 0b1101100
u : 0b1110101
t : 0b1110100
```