**TD6 : Chaîne de caractère**

Le type "chaine de caractère" (string en anglais) est un type dit complexe (puisqu'une chaine de caractère comprend a priori plusieurs caractères). Les données s'écrivent entre apostrophe (simple quote) ou guillemets anglais (double quote).

*Exemple 1* : chaine= "Bonjour" ou 'Bonjour'

*Exemple 2* :

Saisir dans le shell :

chaine= "Bonjour"

len(chaine) la fonction len( ) sert à ……………………………………………….

print(T) la fonction print( ) sert à ……………………………………………….

L'instruction input renvoie par défaut une chaine de caractère

*Exemple 3* :

variable=input("Veuillez saisir quelque chose")

type(variable)

**Concaténation:**

On peut mettre bout à bout deux chaînes de caractères pour en former une nouvelle avec le signe +

*Exemple 4* :

ch1= "Bonjour"

ch2= 'Monsieur Toto'

ch3=ch1+ch2

print(ch3) il s'affiche: ……………………………………………….

L'espace est un caractère de la chaine de caractère.

On ne peut pas concaténer un nombre avec une chaîne de caractère.

*Exemple 5* :

prix=20

ch1= "L'objet coute "

ch2= " euros."

ch3=ch1+prix+ch2

print(ch3) il s'affiche:………………………………. ……………………………………………….

Bien sur on pourrait utiliser print(ch1, prix, ch2). On afficherait ce qui semblerait être une phrase mais serait stockée en différents morceaux (nos trois variables).

La concaténation a pour but de créer une troisième chaine ch3 qui contiendrait toute la phrase.

Pour cela on a besoin de transformer la variable prix (qui contient un entier) en une chaine de caractère grâce à la fonction de transtypage str(). Cette fonction n'existe pas dans tous les langages.

*Exemple 5* *(suite):*

ch3=ch1+str(prix)+ch2

print(ch3) il s'affiche:………………………………. ……………………………………………….

remarque concernant la fonction print: elle sépare a priori les valeurs des variables affichées par des espace et termine par un retour à la ligne.

Pour ne pas aller à la ligne après un affichage, on peut préciser end=" " (pour terminer par un espace) ou end=""(pour ne rien avoir du tout à la fin de l'affichage). Par défaut, on a end="\n" (\n désigne un retour à la ligne). De même, en cas d'affichage multiple, on peut préciser le séparateur avec sep=" ".

**Parcours de chaîne**

Les caractères d'une chaîne sont numérotés par un indice entier (qui commence à 0).

*Exemple 6* :

chaine="Bonjour"

print( chaine[4] ) …………………………………………….

print(chaine[4:]) …………………………………………….

On peut ne peut pas changer un caractère de la chaine :

chaine="Bonjaur"

chaine[4] = "o" ……………………………..

**Exercices:**

**exercice 1:**

a)Que fait l'algorithme suivant?

variables: i,n entiers

reponse booléen

Entrée: chaine chaine de caractères

début:

reponse ← False

n ← longueur(chaine)

Pour i allant de 0 à n-1 répéter:

Si (chaine[i]="a") alors:

reponse← True

finSi

fin Pour

Sortie: reponse

fin

b)Implémentez le en Python dans un script (c'est-à-dire pas dans le shell!)

c)Cet algorithme peut être optimisé car en effet, des que l'un des caractères de la chaîne est "a", il est inutile de continuer à parcourir la chaîne, on sait que la réponse sera True. Modifier donc la boucle en conséquence.

**exercice 2:** Ecrire un script qui compte l'occurrence (c'est-à-dire le nombre) de "a" contenu dans une chaîne.

**exercice 3:** Ecrire un script qui, à partir d'une chaîne donnée, crée une nouvelle chaine en insérant des astérisques. *Exemple:* chaine1="hello" donne chaine2="h\*e\*l\*l\*o"

**exercice 4:**

a)Ecrire un script qui, à partir d'une chaîne donnée, crée une nouvelle chaine en inversant l'ordre des lettres. *Exemple:* chaine1="hello" donne chaine2="olleh"

b)Le transformer en fonction appelée inverse

def inverse (chaine1):

instructions

return chaine2

c)Utiliser cette fonction inverse pour tester si une chaine est un palindrome (c'est-à-dire une chaine qui peut se lire dans les deux sens comme radar, rotor…)