UNIL | Université de Lausanne

Jérémie Cabessa

Informatique I
Introduction à la programmation

TP 02

Le but de ce TP est de vous familiariser avec le traitement des chaînes de caractères et avec les fonctions d'interaction entre vos programmes et des utilisateurs.

Exercice 1: Chaînes de caractères

Les chaînes de caractères sont indexées, et le premier caractère de la chaîne possède l'indice 0. Dans l'exemple ci-dessous, des "sous-chaînes" peuvent être spécifiées en donnant un intervalle de deux indices séparés par ":". Les indices négatifs commencent par la fin de la chaîne.

```
>>> word = "Bonjour"
>>> word[0]                # l'indexation commence par zéro
'B'
>>> word[4]                # cinquième caractère
'o'
>>> word[0:3]              # une sous-chaine
'Bon'
>>> word[3:4]
'j'
>>> word[-1]               # le dernier caractère
'r'
>>> word[-2]               # l'avant dernier caractère
'u'
```

Indices par défaut: si le premier indice est omis, il est mis à zéro par défaut, si le second indice est omis, il vaut la taille de la chaîne par défaut. La taille de chaine peut être récupérée avec la fonction **`len()`**

```
>>> word[:3]      # Les trois premiers caractères
"Bon"
>>> word[3:]      # Tout sauf les trois premiers caractères
"jour"
>>> len(word)     # Taille de la chaine de caractères
7
```

Exemple: indexation d'une chaine de caractères

caractères de la chaine `word`	**B**	**o**	**n**	**j**	**o**	**u**	**r**
index depuis le début	0	1	2	3	4	5	6
index depuis la fin	-7	-6	-5	-4	-3	-2	-1

Exercice 2: Votre premier programme en Python

Soit une équation du second degré: **$ax^2 + bx + c = 0$**. Initialisez les 3 identificateurs **a**, **b** et **c** à des valeurs quelconques. Déterminez ensuite les valeurs de **x** en faisant le moins d'efforts possibles. Vous pouvez afficher les valeurs de **x** en utilisant la fonction **`print`**, mais attention pour afficher les valeurs numériques il faut d'abord les convertir en chaînes de caractères avec la fonction **`str`**.

Attention: le discriminant de l'équation **(b2 - 4ac)** doit être positif.

Astuces: En Python, x puissance y se calcule avec l'expression: **x**y**. La racine carrée d'un nombre, c'est ce nombre à la **puissance 1/2** (soit 0.5). **Sauvegardez** votre programme dans un fichier pour le réutiliser plus tard.

Exercice 3: Votre second programme en Python

Initialisez un identificateur **nom** contenant votre nom et prénom séparés par un espace. Ecrivez vos nom et prénom une centaine fois à l'écran séparés par " * " (espace - étoile - espace). En utilisant des sous-chaînes de l'identificateur **nom**, faites de même avec seulement vos initiales.

Exercice 4: Interagir avec Python

Vous pouvez aisément interagir avec votre programme. Par exemple, afin de ne pas avoir à fixer les valeurs de **a**, **b** et **c** dans le script qui résout les équations du second degré. En effet, il est gênant que vous deviez éditer et sauvegarder le script à chaque fois que vous désirez changer ces valeurs. Il serait nettement plus pratique que le programme vous demande les valeurs à chaque exécution. Pour ce faire, utilisez les fonction `input()` et `raw_input()` de la manière suivante:

```
>>> nom = raw_input("Quel est votre nom? ") #pour les chaines
Quel est votre nom? Homer
>>> age = input("Quel est votre age? ")     #pour les valeurs numériques
Quel est votre age? 38
>>> print nom + " a " + str(age) + " ans"
Homer a 38 ans
```

Cela affichera le message à l'écran et attendra que l'utilisateur entre une valeur puis presse return. La valeur sera alors assignée à l'identificateur à gauche de l'opérateur d'affectation (dans l'exemple ci-dessus, l'identificateur **nom**.). **`input`** est une fonction intelligente qui convertit la chaîne de caractères entrée par l'utilisateur en nombres (entiers ou décimaux) si cela est possible (i.e. la chaîne ne contient que des chiffres et au plus un point). Sinon, il se produira une erreur lors de l'exécution.

4.1 Ecrivez un nouveau script qui utilise les fonctions **`input`** et **`raw_input`** qui demandera à l'utilisateur son nom, son âge et sa taille en mètres et affichera ces informations de manière conviviale. Inspirez-vous de l'exemple ci-dessous.

```
Quel est votre nom ? Jacques
Quel est votre âge ? 25
Quelle est votre taille en mètres ? 1.85
Bonjour Jacques, vous êtes né en 1985 et faites 1.85m ou 185cm
```

4.2 Modifiez le script de l'exercice 2 afin qu'il demande à l'utilisateur les valeurs des identificateurs **a**, **b** et **c** au lieu qu'elles ne soient spécifiées dans le script.

Attention! N'oubliez pas de sauvegarder les fichiers de scripts que vous avez créés aujourd'hui (.py) sur une clé USB ou dans votre WebDoc, si vous voulez être certain de pouvoir les réutiliser ultérieurement.

Exercice 5: Conversion des types de données en Python

Il est parfois nécessaire de convertir des entiers/nombres décimaux en chaînes de caractères et vice-versa. Nous avons déjà vu la fonction **`str()`** qui convertit ce qui est donné en paramètre (quel que soit son type) en une chaîne de caractères.

Dans le sens inverse, lorsqu'on veut passer d'une chaîne de caractères à une valeur numérique, il est important de savoir si l'on parle d'entiers ou de nombres décimaux: "123.45" ne peut pas être converti en un entier! Il existe donc 2 fonctions:

- **`int()`** pour passer d'une chaîne à un entier;
- **`float()`** pour passer d'une chaîne à un nombre décimal;

Ainsi, on peut faire:

```
>>> s1 = "10"
>>> print s1 + 2 #erreur! on ne peut pas ajouter une chaine et un nombre
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: cannot concatenate 'str' and 'int' objects
>>> x = int(s1)
>>> print 2 + x                 # affiche 12
12
>>> s2 = "13.21"
>>> print int(s1)+float(s2)     # affiche 23.21
23.21
```

5.1 Maintenant, écrivez un nouveau script nommé conversions.py qui demande à l'utilisateur l'heure actuelle sous la forme HH:MM:SS. Aidez-vous de la fonction **`raw_input()`**. Le script doit initialiser 3 identificateurs **hs**, **mins** et **secs** avec les paires de caractères correspondantes aux heures, minutes et secondes et puis les convertir en entiers via les identificateurs **hint**, **minint** et **secint** qui seront utilisés pour calculer le nombre des secondes après minuit.

Par exemple:

```
Entrez l'heure a convertir: 10:16:35
10:16:35 vaut 36995 secondes
```

5.2 Modifiez le code afin que la conversion de secondes se fasse en nombres décimaux, i.e. pour qu'il comprenne le format: HH:MM:SS.SSS