Universidad Peruana Cayetano Heredia
Facultad de Ciencias y Filosofía
Departamento de Ciencias Exactas

Computación 2022 Verano

Lab. 4.2. Procesamiento de String en Python. Parte III

Enero 24, 2022

Cree una carpeta (folder) en el disco D, nómbrela con su apellido paterno seguido de su código. Ejemplo: LOPEZ12345

El conjunto de los símbolos (caracteres) del código ASCII.

Todos los símbolos que se usan en computación normalmente están agrupados y ordenados bajo el sistema ASCII. A cada símbolo (carácter) le corresponde un número ordinal, es decir que cada símbolo del sistema (código) ASCII tiene asignado un número que lo representa.
El sistema (código) ASCII tiene 256 símbolos o caracteres.

Si queremos averiguar el código ASCII de la letra "A", usamos la función **ord( )** de la siguiente manera:

```
>>> print(ord('A'))
65
```

Si queremos averiguar el símbolo o carácter que le corresponde al número 64, usamos la función **chr( )** de la siguiente manera:

```
>>> print(chr(64))
@
```

Procesamiento de cadenas de caracteres (string)

Un **string** es un conjunto de caracteres ordenados en forma secuencial y que conforman una unidad. Eso quiere decir que cada elemento del **string** tiene asignado un número de orden o dicho de otra manera, tiene asignado su **índice**. La asignación de los índices lo realiza Python en forma automática.

Python asigna el índice empezando en **cero**, luego aumentando en 1 en forma secuencial de izquierda a derecha. Sin embargo, Python también permite asignar índices negativos de derecha a izquierda empezando en -1.

Índice	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	15	17	18	19
Cadena:	**H**	**o**	**l**	**a**		**c**	**o**	**m**	**o**		**e**	**s**	**t**	**a**	**s**	**,**	**J**	**o**	**s**	**e**
Índice:	-20	-19	-18	-17	-16	-15	-14	-13	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

Todo **string** es inmutable, es decir que no se puede modificar su contenido. Si se desea modificar, lo que se hace es generar otro **string** con las modificaciones esperadas.

Cómo acceder a un elemento de una cadena de caracteres (string)

Para acceder a un elemento de un string basta con colocar el nombre del string y entre corchetes su índice:

```
>>> A="casa"
>>> print(A[2])
's'
```

```
>>> miString = "hola amigo"
>>> print(miString[-4])
'm'
```

Operaciones básicas

- El operador **+** (concatenación de cadenas): acepta dos cadenas como operadores y devuelve la cadena que resulta de unir la segunda a la primera.

```
>>> 'abc' + 'def'
'abcdef'
```

```
>>> x="hola"
>>> y="amigo"
>>> x+y
"holaamigo"
```

- ¿Qué sucede si queremos concatenar un string con un número? Pruebe lo siguiente:

```
>>> print(4 + "you" + 2 + "more")
```

- La función `str()` permite convertir un número a string.

```
>>> print(str(4) + "you" + str(2) + "more")
```

- Operador ***** (repetición de cadena): acepta una cadena y un entero y devuelve la concatenación de la cadena consigo misma tantas veces como indica el entero.

```
>>> "N1"*5
'N1N1N1N1N1'
```

```
>>> x="hola"
>>> x*4
"holaholaholahola"
```

- La función len(), devuelve la cantidad de caracteres que contiene la cadena.

```
>>> len("hola! Dr.")
9
>>> s="mi mamá me mima"
>>> len(s)
15
```

¡¡Cuidado!!. El índice del último elemento no es 9

Como imprimir los caracteres de una cadena uno a uno

Podemos imprimir uno a uno los caracteres de una cadena de dos maneras, a)-recorrido por índice, b)-recorrido por valor. En esta parte del curso usaremos principalmente el recorrido por índice.

a):

```
s= "mi mamá me mima"
x= len(s)
for i in range(0,x):
      print(s[i],end="")
```

b):

```
s= "mi mamá me mima"
for i in s:
      print(i,end="")
```

Para cada uno de los siguientes ejercicios implemente el programa Python correspondiente. Guarde vuestros programas en vuestra carpeta de trabajo.

1. Escriba un programa que lea un texto del teclado e imprima ése mismo texto, pero con las palabras en orden contrario.

```
Ingrese un texto: Estoy seguro que aprobé el curso
curso
el
aprobé
que
seguro
Estoy
```

2. Implemente un programa que reciba una frase y devuelva la misma frase eliminando los espacios sobrantes, de tal forma que entre palabra y palabra sólo quede un espacio. El programa debe de eliminar también los espacios en blanco del inicio y del final de la frase.

```
Ingrese una frase:    El examen N° 3   es el    miercoles 10 de  noviembre
Frase modificada:  El examen N° 3 es el miercoles 10 de noviembre
```

3. Escriba un programa que ingrese un texto del teclado y determine si el texto contiene todas las letras del alfabeto (ingles). Ejemplo de salida:

```
Ingrese una frase:
Le gustaba cenar un exquisito sandwich de jamon con zumo de pera y VODKA fria

La frase contiene todas las letras del alfabeto

Ingrese una frase:
The quick brown fox jumps over the lazy dog

La frase contiene todas las letras del alfabeto

Ingrese una frase:
The quick brown fox jumps over the lazy cat

La frase no contiene todas las letras del alfabeto, faltan: d g
```

4. Un anagrama es una palabra que se forma por la combinación de letras de otra palabra o frase. Tienen los mismos caracteres con la misma frecuencia de apariciones, pero en orden diferente. Ejemplos:

```
CALLAO - LACOLA
ROMA - AMOR - OMAR - MORA - RAMO
ROLDAN - LADRON
MONJA - JAMON
```

   Escriba un programa que determine si dos palabras son anagramas

5. Escriba un programa que imprima en orden lexicográfico todas las letras comunes de dos frases cuyas letras están en minúsculas.
   Ejemplo de salida:

```
Ingrese la primera frase: me gusta python
Ingrese la segunda frase: me gusta programar

Las letras comunes son: a e g m o p s t u

Ingrese la primera frase: i love R
Ingrese la segunda frase: why?

No existen letras comunes
```

Guarde todos vuestros programas y vuestra hoja de respuestas en una carpeta con el nombre su **Apellido** paterno seguido de vuestro **DNI**, luego comprima esta carpeta. Envíe este archivo a victor.melchor.e@upch.pe , especificando como asunto **Lab4.2**.