IIC1103 - TAV

Clase 6 módulo 2: Métodos de strings

Prof. Francisca Cattan

Universidad Católica de Chile

11 de enero, 2023

¿Que sabemos hasta ahora?

1. Variables y expresiones.
2. Control condicional
3. Control de flujo
4. Funciones
5. Recursión
6. Strings
   - Definir strings.
   - Imprimir strings.
   - Concatenar strings.
   - Conversión de strings.
   - Formatear strings.
   - Obtener trozos o *slices* de strings.
   - Recorrer strings como secuencias.

En el módulo pasado: definir strings

El caracter "\" permite poner comillas, saltos de línea y tabs dentro de un string.

Secuencia	Significado
\'	Comilla simple
\"	Comilla doble
\n	Salto de línea
\t	Tabulador
\\	Backslash

En el módulo pasado: definir strings

```
1 s = "Luke ... \n\t \"yo soy tu padre\""
2 print(s)
```

```
Luke ...
        "yo soy tu padre"
```

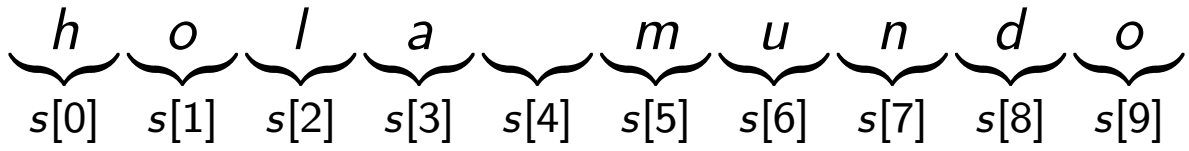
En el módulo pasado: formatear strings

Python permite convertir un número a un string en diversos formatos. El método `format` permite completar los de un `str`.

```
1  t = "En primer lugar queda {} y le sigue {}"
2  s = t.format("Leclerc", "Perez")
3  print(s)
4
5  t = "{0} gana con {1} puntos y {3} con {2} puntos"
6  s = t.format("Leclerc","26","18","Perez")
7  print(s)
8
9  t = "La diferencia fue de {0:.2f}s"
10 s = t.format(20.524312)
11 print(s)
```

```
En primer lugar queda Leclerc y le sigue Perez
Leclerc gana con 26 puntos y Perez con 18 puntos
La diferencia fue de 20.52s
```

En el módulo pasado: strings como secuencias



Probemos obtener trozos o *slices* del string.

```
1 s = "hola mundo"
2 print(s[1])
3 print(s[3:8])
4 print(s[:5])
5 print(s[8:])
6 print(s[3:8:3])
7 print(s[8:1:-1])
8 print(s[::-1])
```

```
o
a mun
hola
do
au
dnum al
odnum aloh
```

En el módulo pasado: strings como secuencias

Podemos encontrar el largo del sring usando la función **`len()`**. También podemos recorrer los caracteres de un string usando la instrucción **`for`**.

```
1 s = "hola mundo"
2 for i in range(len(s)): # usamos len() para usar como rango
3     print(s[i])
```

```
1 s = "hola mundo"
2 for i in s: #recorremos el str completo por cada caracter
3     print(i)
```

En el módulo pasado: strings como secuencias

Usando el operador lógico **`in`** podemos recorrer el string. También podemos verificar si un caracter pertenece a un string o no.

```
1 s = "hola mundo"
2 for i in s:
3     if i in 'aeiou':
4         print('vocal encontrada!!')
```

Strings

Hagamos un ejercicio con lo que sabemos hasta ahora:

- Escriba una función que reciba un string **s** y retorne **True** ssi **s** es un palíndromo. Un palíndromo es una palabra que se lee igual en ambos sentidos sin considerar espacios.

Strings

Hagamos un ejercicio con lo que sabemos hasta ahora:

- Escriba una función que reciba un string **s** y retorne **True** ssi **s** es un palíndromo. Un palíndromo es una palabra que se lee igual en ambos sentidos sin considerar espacios.

Separemos en pasos:

1. Quitar los espacios.
2. Invertir la palabra.
3. Verificar si la palabra original es igual a la palabra invertida.

Strings

Resolver un detector de palíndromos:

```
1 def quitar_espacios(s): # retorna "s" sin espacios
2     nuevo_s = ""
3     for c in s:
4         if(c != " "):
5             nuevo_s += c
6     return nuevo_s
```

Strings

Resolver un detector de palíndromos:

```
1 def quitar_espacios(s): # retorna "s" sin espacios
2     nuevo_s = ""
3     for c in s:
4         if(c != " "):
5             nuevo_s += c
6     return nuevo_s
```

```
1 def invertir(s): # retorna "s" invertido
2     nuevo_s = ""
3     for c in s:
4         nuevo_s = c + nuevo_s
5     return nuevo_s
```

Strings

Resolver un detector de palíndromos:

```
1 def quitar_espacios(s): # retorna "s" sin espacios
2     nuevo_s = ""
3     for c in s:
4         if(c != " "):
5             nuevo_s += c
6     return nuevo_s
```

```
1 def invertir(s): # retorna "s" invertido
2     nuevo_s = ""
3     for c in s:
4         nuevo_s = c + nuevo_s
5     return nuevo_s
```

```
1 def palindromo(s):
2     s = quitar_espacios(s)
3     return s == invertir(s) # que tipo de dato retorna?
```

Strings

Resolver un detector de palíndromos:

```
1 def quitar_espacios(s): # retorna "s" sin espacios
2     nuevo_s = ""
3     for c in s:
4         if(c != " "):
5             nuevo_s += c
6     return nuevo_s
```

```
1 def invertir(s): # retorna "s" invertido
2     nuevo_s = ""
3     for c in s:
4         nuevo_s = c + nuevo_s
5     return nuevo_s
```

```
1 def palindromo(s):
2     s = quitar_espacios(s)
3     return s == invertir(s) # que tipo de dato retorna?
```

```
1 # llamamos a la funcion con un string cualquiera
2 print(palindromo("yo soy tu padre"))
3 print(palindromo("yo hago yoga hoy"))
```

Funciones para la clase String

Existen muchos métodos especiales para `str`. La sintaxis será similar a la función `format()`:

Sintaxis de métodos para `str`

```
string.metodo(argumento1,argumento2,...)
```

Es muy importante entender que al string se le aplica el método, según los parámetros entregados. Dependiendo del método, el resultado será información sobre el string, o una nueva variable correspondiente al string modificado.

Funciones para la clase String: mayúsculas y minúsculas

- `s.lower()`: Retorna `s` en minúscula.
- `s.upper()`: Retorna `s` en mayúsculas.
- `s.capitalize()`: Retorna `s` con la primera letra en mayúscula.

```
1 s = "Cuando se juega al Juego de Tronos, solo se puede
  ganar o morir."
2 s = s.upper()
3 print(s)
4 s = s.lower()
5 print(s)
6 s = s.capitalize()
7 print(s)
```

```
CUANDO SE JUEGA AL JUEGO DE TRONOS, SOLO SE PUEDE GANAR O MORIR.
cuando se juega al juego de tronos, solo se puede ganar o morir.
Cuando se juega al juego de tronos, solo se puede ganar o morir.
```

Funciones para la clase String: mayúsculas y minúsculas

- `s.lower()`: Retorna `s` en minúscula.
- `s.upper()`: Retorna `s` en mayúsculas.
- `s.capitalize()`: Retorna `s` con la primera letra en mayúscula.

```
1 s = "Cuando se juega al Juego de Tronos, solo se puede
  ganar o morir."
2 s = s.upper()
3 print(s)
4 s = s.lower()
5 print(s)
6 s = s.capitalize()
7 print(s)
```

Obs: En esta sintaxis, el retorno de la función será asignado a la variable `s`. Esto quiere decir que el string `s` no se modifica permanentemente por aplicarle el método.

Funciones para la clase String: ¿es mayúscula?¿es minúscula?

- `s.isupper()`: Retorna True si todos los caracteres de `s` están en mayúsculas.
- `s.islower()`: Retorna True si todos los caracteres de `s` están en minúsculas.

```
1 s = "NO SABES NADA JON SNOW"
2 print(s.isupper())
3 print(s.islower())
```

```
True
False
```

Funciones para la clase String: buscar caracteres

- `s.find(c)`: Retorna el índice de la primera ocurrencia de `c` en `s`. Si no existe retorna -1.
- `s.rfind(c)`: Lo mismo, pero busca de derecha a izquierda.

```
1 s = "Cuando se juega al Juego de Tronos, solo se puede
  ganar o morir."
2 print(s.find("a"))
3 print(s.rfind("a"))
4 print(s.find("Juego"))
5 print(s.rfind("perder"))
```

```
2
53
19
-1
```

Funciones para la clase String: remover en extremos

- `s.strip(c)`: Retorna `s` sin `c` sólo a ambos costados.
- `s.lstrip(c)`: Retorna `s` sin `c` a la izquierda.
- `s.rstrip(c)`: Retorna `s` sin `c` a la derecha.

```
1 s = "\t poder es poder"
2 print(s.strip("poder"))
3 print(s.lstrip()) # quita \n \t y espacios al inicio
4 s = "poder es poderpoder"
5 print(s.strip("poder"))
```

```
 poder es
poder es poder
 es
```

Funciones para la clase String: prefijos y sufijos

- `s.startswith(c)`: Retorna True ssi `s` comienza con `c`.
- `s.endswith(c)`: Retorna True ssi `s` termina con `c`.

```
1 s = "Nunca olvides lo que eres, el resto del mundo no lo
  hara. Llevalo como una armadura y nunca lo usaran para
  herirte."
2 print(s.startswith("Nunca olvid"))
3 print(s.endswith("erirte."))
```

```
True
True
```

Funciones para la clase String: ¿letras o números?

- `s.isalpha()`: Retorna True ssi `s` sólo tiene letras.
- `s.isdigit()`: Retorna True ssi `s` sólo tiene dígitos.

```
1 s = "Un lector vive 1000 vidas antes de morir. Aquel que
  nunca lee vive solo 1."
2 print(s.isalpha())
3 print(s[3:9].isalpha()) # s[3:9] = lector
4 print(s[15:19].isdigit()) # s[15:19] = 1000
```

```
False
True
True
```

Funciones para la clase String: ¿cuántas veces?

- `s.count(c)`: Retorna la cantidad de ocurrencias de `c` en `s`.

```
1 s = "El Norte no olvida."
2 print(s.count("l"))
3 print(s.count("o"))
4 print(s.count("Norte"))
5 print(s.count("Sur"))
```

```
2
3
1
0
```

Funciones para la clase String: reemplazar

- `s.replace(`c_1`,` c_2`)`: Retorna un string igual a `s`, pero reemplazando cada ocurrencia de c_1 por c_2.

```
1  s = "La guerra es entre los vivos y los muertos. No te
   equivoques, ¡los muertos estan en camino!"
2
3  # cambia una frase por otra
4  print(s.replace("La guerra", "Esta paz"))
5  print(s.replace("muertos", "MU3R70Z"))
6  # quita los espacios
7  print(s.replace(" ", ""))
8  # quita los signos de puntuacion
9  s = s.replace(".", "").replace("¡", "").replace("!", "")
10 print(s)
```

```
Esta paz es entre los vivos y los muertos.  No te equivoques,
¡los muertos estan en camino!
La guerra es entre los vivos y los MU3R70Z. No te equivoques,
¡los MU3R70Z estan en camino!
Laguerraesentrelosvivosylosmuertos.Noteequivoques,¡losmuertosestanenca
La guerra es entre los vivos y los muertos No te equivoques, los
muertos estan en camino
```

Strings

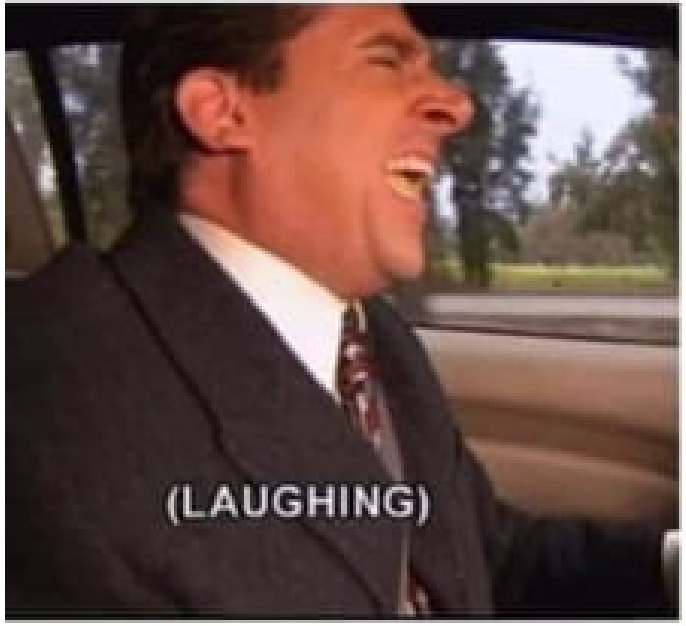
Veamos algunos ejemplos:

- Escriba una función que reciba un string y lo retorne en minúscula y sin signos de puntuación (.,;-/¡!¿?) ni caracteres espaciales (\n, \t, \", \', \\)
- Escriba un reconocedor de palíndromos que no considere ni espacios ni signos de puntuación.
- Escriba una función que reciba un string `s` y retorne True ssi `s` es un correo electrónico.

Strings

Veamos más ejemplos!! :D:

- Escriba una función que reciba un string `s` y retorne True ssi `s` puede ser convertido en float.
- Escriba una función que reciba un string `s` y retorne la cantidad de números enteros que en él aparecen.
- Cree una función que reciba `s` y retorne True ssi `s` es un pangam. Un pangam contiene todas las letras del alfabeto sin considerar eñes ni tildes.



Looking at programming memes



Actually coding