Métodos de formato

`capitalize()` nos permite devolver la cadena con el primer carácter en mayúsculas.

```
>>> cad = "hola, como estás?"
>>> print(cad.capitalize())
Hola, como estás?
```

`lower()` y `upper()` convierte la cadena de caracteres en minúsculas y mayúsculas respectivamente.

```
>>> cad = "Hola Mundo"
>>> print(cad.lower())
hola mundo

>>> cad = "hola mundo"
>>> print(cad.upper())
HOLA MUNDO
```

`swapcase()`: devuelve una cadena nueva con las minúsculas convertidas a mayúsculas y viceversa.

```
>>> cad = "Hola Mundo"
>>> print(cad.swapcase())
hOLA mUNDO
```

`title()`: Devuelve una cadena con los primeros caracteres en mayúsculas de cada palabra.

```
>>> cad = "hola mundo"
>>> print(cad.title())
Hola Mundo
```

Métodos de búsqueda

`count()`: Es un método al que indicamos como parámetro una subcadena y cuenta cuantas apariciones hay de esa subcadena en la cadena.

```
>>> cad = "bienvenido a mi aplicación"
>>> cad.count("a")
3
```

Además podemos indicar otro parámetro para indicar la posición desde la que queremos iniciar la búsqueda. Y otro parámetro optativo para indicar la posición final de búsqueda.

```
>>> cad.count("a",16)
2
>>> cad.count("a",10,16)
1
```

`find()` nos devuelve la posición de la subcadena que hemos indicado como parámetro. Sino se encuentra se devuelve -1.

```
>>> cad.find("mi")
13
>>> cad.find("hola")
-1
```

Métodos de validación

`startswith()` nos indica con un valor lógico si la cadena empieza por la subcadena que hemos indicado como parámetro. Podemos indicar también con otro parámetro la posición donde tiene que buscar.

```
>>> cad.startswith("b")
True
>>> cad.startswith("m")
False
>>> cad.startswith("m",13)
True
```

`endswith()` igual que la anterior pero indica si la cadena termina con la subcadena indicada. En este caso, se puede indicar la posición de inicio y final de búsqueda.

```
>>> cad.endswith("ción")
True
>>> cad.endswith("ción",0,10)
False
>>> cad.endswith("nido",0,10)
True
```

Otras funciones de validación: `isdigit()`, `islower()`, `isupper()`, `isspace()`, `istitle()`,...

Métodos de sustitución

`replace()`: Devuelve una cadena donde se ha sustituido las apariciones de la primera subcadena indicada por la segunda subcadena indicada como parámetro.

```
>>> buscar = "nombre apellido"
>>> reemplazar_por = "Juan Pérez"
>>> print ("Estimado Sr. nombre apellido:".replace(buscar, reemplazar_por))
Estimado Sr. Juan Pérez:
```

`strip()`: Devuelve una cadena donde se han quitado los espacios del principio y del final. Si indicamos una subcadena como parámetro quitará dicha subcadena del principio y del final.

```
>>> cadena = "   www.eugeniabahit.com   "
>>> print(cadena.strip())
www.eugeniabahit.com
```

```
>>> cadena="0000000123000000000"
>>> print(cadena.strip("0"))
123
```

Métodos de unión y división

aunque todavía no lo hemos estudiado, el método `split()` nos permite convertir una cadena en una lista. Lo usaremos más adelante.

```
>>> hora = "12:23:12"
>>> print(hora.split(":"))
['12', '23', '12']
```

`splitlines()`: Nos permite separar las líneas que hay en una cadena (indicada con el carácter `\n`) en una lista.

```
>>> texto = "Linea 1\nLinea 2\nLinea 3"
>>> print(texto.splitlines())
['Linea 1', 'Linea 2', 'Linea 3']
```