Clase Nº 6b

Expresiones Regulares

Concepto

- Las ***expresiones regulares*** son secuencias de caracteres que forman un patrón de búsqueda, lo que permite encontrar coincidencias y contar sus repeticiones en documentos y todo tipo de textos.

Concepto

- También sirven para detectar formatos determinados, como direcciones de correo electrónico, códigos de inventario, patentes vehiculares, etc.
- Python dispone de varias funciones para trabajar con ellas, todas incluidas en el módulo ***re***.

Función search( )

- **re.search(<patrón>, <cadena>):** Busca la primera coincidencia de *<patrón>* dentro de *<cadena>*. Devuelve un objeto *re.Match*.

Función search( )

```
cad = "La bella y graciosa moza se marchó a lavar la ropa"
resultado = re.search("moza", cad)
print(resultado)
```

```
<re.Match object; span=(20, 24), match="moza">
```

- Devuelve *None* si no hay coincidencia.
- Puede añadirss re.IGNORECASE para ignorar mayúsculas y minúsculas.

Objeto Match

- Con los métodos *start* y *end* se puede averiguar dónde comienza y termina la coincidencia:

```
print(resultado.start())  # 20
print(resultado.end())    # 24
```

Función findall( )

- **re.findall(<patrón>, <cadena>): Busca la <u>todas</u> las coincidencia de <*patrón*> dentro de <*cadena*>. Devuelve una lista con las coincidencias halladas, o una lista vacía si no se encuentra.**

Función findall( )

```
cad = "La bella y graciosa moza se marchó a lavar la ropa"
resultado = re.findall("la", cad)
print(resultado)    # ['la', 'la', 'la']
```

Metacaracteres

- Los ***metacaracteres*** o ***comodines*** se utilizan para representar patrones de búsqueda.
- Su significado es interpretado según el metacarácter utilizado.

Metacarácter .

- El punto representa a *cualquier carácter* dentro de la cadena:

```
cad = "La bella y graciosa moza se marcho a lavar la ropa"
resultado = re.findall(".o.a", cad)
print(resultado)    # ['iosa', 'moza', 'ho a', 'ropa']
```

Metacarácter ^

- El tilde circunflejo (^) o *ancla de inicio* indica que el patrón debe buscarse sólo al comienzo de la cadena:

```
cad = "La bella y graciosa moza se marchó a lavar la ropa"
lista = cad.split()
for palabra in lista:
    if re.findall("^l", palabra):
        print(palabra)    # lavar la
```


Metacarácter $

- El signo pesos ($) o *ancla de fin indica* que el patrón debe buscarse sólo al final de la cadena:

```
cad = "La bella y graciosa moza se marcho a lavar la ropa"
lista = cad.split()
for palabra in lista:
    if re.findall("o$", palabra):
        print(palabra)    # marcho
```

Grupos de caracteres

- Cuando se encierra un grupo de caracteres entre corchetes se busca coincidencia con cualquiera de ellos:

```
cad = "Después de lavar la ropa la niña tomó la sopa"
lista = re.findall("[rs]opa", cad)
print(lista)    # ['ropa', 'sopa']
```


Rangos de caracteres

- Con un guión dentro de un grupo de caracteres se puede definir un rango (desde-hasta).
  - [0-9] equivale a [0123456789]
  - [a-d] equivale a [abcd]
  - [P-Z] equivale a [PQRSTUVWXYZ]

Anclas en grupos y rangos

- Los anclas (^ y $) se pueden combinar libremente con los grupos y rangos:

```
cad = "La bella y graciosa moza se marchó a lavar la ropa"
lista = cad.split()
for palabra in lista:
    if re.search("^[b-k]", palabra):
        print(palabra)    # bella graciosa
```


Cuantificadores

- Los cuantificadores son símbolos que indican cuantas veces debe aparecer en el texto el carácter que le antecede.
  - ?: Indica 0 o 1 vez
  - *: indica 0 o más veces
  - +: indica 1 o más veces

Cuantificadores

```
lista = ["12","102", "1002", "10002"]

for numero in lista:
    if re.search("10?2", numero):
        print(numero)   # 12 102

for numero in lista:
    if re.search("10+2", numero):
        print(numero)   # 102 1002 10002
```


Función sub( )

- re.sub(<patrón>, <reemplazo>, <cadena>: Busca la todas las coincidencia de <*patrón*> dentro de <*cadena*> y las reemplaza por <reemplazo>- Devuelve una nueva cadena modificada, o la misma si no se encuentra.

Función sub( )

```
cad = "La bella y graciosa moza se marcho a lavar la ropa"
cad2 = re.sub("bella.* la", "[...]", cad)
print(cad2)   # La [...] ropa
```


¡A practicar!