Clase 6 Cadenas II

 Cadenas de Caracteres. Métodos. Formateo de cadenas

Metodos para cadenas

- Los nombres de los siguientes metodos se caracterizan por comenzar con la particula **is**
- Se trata de metodos interrogativos, que no llevan parametros y permiten obtener informacion acerca de una cadena.
- Todos los metodos que comienzan con is devuelven **True** o **False**

- <str>.isalpha():Reconoce caracteres regionales.
- True → son letras
- False → en caso contrario

```
cad = "Hola"
print(cad.isalpha()) #True
```

- <str>.isdigit(): Devuelve True si todos los caracteres de <str> son digitos numericos.
- es natural, entero, positivo o cero

```
cad1 = '1234'
cad2 = '3.1416'
print(cad1.isdigit(),cad2.isd
igit()) #True False
```

- <str>.isalnum(): Devuelve True si todos los caracteres de <str> son alfabeticos o numericos.

```
cad1 = '-1234'
cad2 = 'XR150'
```

```
print(cad1.isalnum( ), cad2.i
salnum( )) #False True
```

- <str>.isupper(): Devuelve True si todos los caracteres alfabeticos de <str> estan en mayuscula. Ignora los caracteres no alfabeticos.

```
cad = 'Lima 717'
if cad.isupper():
    print('Esta en mayuscul
a')
```

- <str>.islower(): Devuelve True si todos los caracteres alfabeticos de <str> estan en minusucula. Ignora los caracteres no alfabeticos.

```
cad = 'azucar: 150 gramos'
print(cad.isslower()) #True
```

EJEMPLO 1

Escribir una funcion para separar una palabra de los signos de puntuacion que pueda tener , devolviendo tres strings : prefijo , palabra , sufijo.

```
#cad = "(ingeniero)"
#limpiarpalabra(cad) #(',ingeniero,')
#limpiarpalabra("increible!") ",increible,!"

def limpiarpalabra(palabra):
    # Inicializa el índice i para buscar el inicio de la pala
bra
    i = 0
    # Avanza i hasta encontrar el primer carácter alfabético
    while i < len(palabra) and not palabra[i].isalpha():
        i = i + 1

    # Almacena los caracteres antes del inicio de la palabra
    inicio = palabra[:i ]

    # Inicializa el índice j para buscar el final de la palab
```

```
ra
    j = len(palabra) - 1

    # Retrocede j hasta encontrar el último carácter alfabéti
co
    while j > i and not palabra[j].isalpha():
        j = j - 1

    # Almacena los caracteres después del final de la palabra
    final = palabra[j + 1:]

    # Extrae la palabra de la cadena original
    palabra = palabra[i:j + 1]

    # Devuelve las tres partes: inicio, palabra y final
    return inicio, palabra, final

# Solicita al usuario que ingrese una palabra para filtrar
palabra = input("Ingrese una palabra para filtrar: ")
# Aplica la función limpiarpalabra a la palabra ingresada
filtracion = limpiarpalabra(palabra)
# Imprime el resultado de la función
print(filtracion)
```

- <str>.upper(): Devuelve <str> convertida a mayuscula. Los caracteres no alfabeticos no resultan modificados.

```
cad = 'hola 123'.upper()
print(cad) #HOLA 123
```

- <str>.lower(): Devuelve <str> convertida a minuscula. Los caracteres no alfabeticos no resultan modificados.

```
cad = 'HOLA 123'.upper()
print(cad) #hola 123
```

- <str>.capitalize(): Devuelve <str> convertiendo a mayuscula el primer caracter de la cadena, y todo lo demas a minusculas.

```
cad = 'Lunes MARTES'
cad = cad.capitalize()
print(cad) #Lunes martes
```

> str→ no empieza con letra no la hace la mayus

- <str>.tittle(): se pone en mayúsuclas la primera letra de cada palabra.
- Detecta la primera letra de cada palabra automáticamente

```
cad = 'Lunes MARTES'
cad = cad.capitalize()
print(cad) #Lunes Martes
```

EJEMPLO 2

Ingresar por teclado el nombre de una entidad o institución y generar la sigla correspondiente tomando la inicial de cada una de sus palabras.

```
#Estrategia 1
cad = input("Ingrese una cadena:")
listadepalabras = cad.split()
sigla = ""
for palabra in listadepalabras:
    sigla = sigla + palabra[0].upper()
print(sigla)

#Estrategia 2
cad = input("Ingrese una cadena:")
sigla = "".join([p[0] for p in cad.split()].upper())
print(sigla)
```

- <str>.ljust(<ancho>[,<relleno>]): Devuelve <str> alineado a la
- <str>.rjust(<ancho>[,<relleno>]): Devuelve <str> alineada a la

izquierda en el ancho especificado. El final de la cadena se rellena con espacios o con el caracter de relleno, si esta presente.

```
cad1 = 'Hola'
cad2 = cad1.ljust(10,'-')
print(cad2) #Hola------
```

derecha. El comienzo de la cadena se rellena con espacios o con el caracter de relleno, si esta presente.

```
cad1 = 'Hola'
cad2 = cad1.rjust(10,'-')
print(cad2) #------Hola
```

- <str>.center(<ancho>[,<relleno>]): Devuelve <str> centrado en el ancho especificado. El resto de la cadena se rellena con espacios o con el caracter de relleno, si esta presente.

```
cad1 = 'Hola'
cad2 = cad1.center(10,'-')
print(cad2) #---Hola---
```

- <str>.zfill(<ancho>): Devuelve <str> alineada a la derecha en el ancho especificado. El comienzo de la cadena se rellena con ceros.

```
n = 3
cad = str(n).zfill(5)
print(cad) #00003
```

facturas

- <str>.rstrip([<cad>]) : Elimina cualquiera de los caracteres de <cad> del final de <str>. Si <cad> se omite se eliminaran los espacios.

```
a = 'Continuara...'
b = a.rstrip('.') #'Continuar
a'
```

- <str>.lstrip([<cad>]) : Elimina cualquiera de los caracteres de <cad> del inicio de <str>. Si <cad> se omite se eliminaran los espacios.

```
a = '--> Salida -->'
b = a.lstrip('- >') #Salida -
->

#se frena en el espacio no en
S
```

- <str>.strip([<cad>]): Elimina cualquier de los caracteres de <cad> de ambos extremos de <str>. Si <cad> se omite eliminaran los espacios.

  > ELIMINA AMBOS LADOS

```
lista = [1,2,3,4]
cad1 = str(lista) #[1,2,3,4]
cad2 = cad1.strip("[]") #1,2,
3,4
```

EJEMPLO 3

Desarrollar un programa para imprimir una factura de venta para una empresa alimenticia

```
print("-"*40)
print("Industrias Alimenticas
S.A".center(40))
print()
print("Avenida del Campo 27
9".ljust(20),end="")
print("Tel. 46019-1146".rjust
(20))
print()
numero = 12
print("Factura N°",str(numer
o).zfill(8))
print()
articulo = "94156 Aceite de o
livia premium"
print(articulo.lstrip("012345
6789").ljust(30,'.'),end="")
precio = 416
print(str(precio).rjust(1
0,'.'))
print("-"*40)
```

```
>>> %Debug -c $EDITOR_CONTENT
----------------------------------------
      Industrias Alimenticias S.A

Avenida del Campo 279       Tel 1573678385

Facrtura N° 00000012

Aceite de oliva premium.............416
----------------------------------------
>>>
```

Métodos para cadenas

- <str>.find(<cad>[,<inicio>[,<fin>]]): Busca la primera aparicion de <cad> dentro de <str>. Devuelve la posicion donde se encontro. A diferencia de index, find no provoca un error si no se encontro, sino que devuelve -1. Pueden indicarse los subindices donde comenzara y terminara la busqueda.
- <str>.rfind(<cad>[,<inicio>[,<fin>]]): Similar a find, pero busca la ultima aparicion de <cad> dentro de <str>. Pueden indicarse los subindices donde comenzara y terminara la busqueda. Si no se los detalla se asumen los extremos de <str>.

  → desde el final de la cadena, busca la última aparición de la cadena

F-strings

- Cadenas formateadas que proporcionan una manera simplificada para darle formato a los datos, **muy superior al operador %.**
- Consisten en cadena normales precedidas por la letra f.
- En los lugares donde deben insertarse los datos se escriben las variables correspondiente, encerradas entre llaves. A estas llave se las denomina **marcadores de posicion**.

```
dia = 16
mes = "Enero"
cad = f"Sucedio el {dia} de {mes}"
print(cad) #Sucedio el 16 de Enero.
```

- Tambien pueden escribirse expresiones en lugar de variables.

```
print(f"{2*37}") #74
```

- O invocarse funciones y metodos.

```
dia = "Lunes"
print(f"{dia.lower()}") #l
```

```
unes
```

- Para alinear numeros y cadena se puede colocar un numero largo del nombre de la variable, separado por “dos puntos”. Este numero representara el ancho.

```
num = 15
cad = f"{num:5}*" #ancho del valor
print(cad) #15*
```

- Escribiendo un 0 delante del ancho se rellena con ceros en lugar de espacios. Solo funciona con el caracter 0.

```
num = 15
cad = f"{num:05}*"
print(cad) #00015*
```

- En forma predeterminada los numeros se alinean a la derecha y las cadenas a la izquierda. Esto puede alterarse escribiendo **<**,**>** o **^** entre los dos puntos y el ancho
  - <: Fuerza alineación a la izquierda
  - >: Fuerza alineación a la derecha
  - ^: Fuerza alineación en el centro

```
#Ejemplos
num = 15
pal = "Hola"
#Alinear un numero a la izqui
erda:
cad = f"*{num:<8}"    #*15
*
#Alinear un numero a la derec
ha:
cad = f"*{pal:>8}"    #*    H
ola*
#Centrar un string
cad = f"*{pal:^8}*"   #*    H
ola   *
```

- Cuando se utilizan los simbolos anteriores es posible indicar cual sera el caracter de relleno. Este

```
pal = "Hola"
cad = f"*{pal:=^8}*" #*==Hola
```

caracter se escribe entro los “dos puntos” y el simbolo de alineacion.

```
==*
```

- Para regular la cantidad de decimlaes de un numero real se escribe un punto, la cantidad de decimales deseada y una letra f detras de los “dos puntos”

```
pi = 3.1416
cad = f"{pi:2f}"
print(cad) #3.14
```

EJEMPLO 4

Imprimir un triangulo de Floyd, ingresando la cantidad de filas

```
n = int(input("Cantidad de filas :"))
k = 1
for i in range(n):
    for j in range(i+1):
        print(f"{k:3}",end="")
        k = k+1
    print()
```

#consola