MÁS TIPOS ESTRUCTURADOS

CADENAS Y DICCIONARIOS

Fundamentos de Informática

Grado en Ingeniería Química

Escuela Técnica Superior de Ingenieros Industriales y de Telecomunicación

UC | ETS de Ingenieros Industriales y de Telecomunicación

MÁS TIPOS ESTRUCTURADOS: CADENAS I

Una **cadena** es una sucesión de caracteres, un vector de caracteres.

Ejemplo. Sea la cadena:

>>> cad1='Hola a todos'

cad1	H	o	l	a		a		t	o	d	o	s
Index positiva	0	1	2	3	4	5	6	7	8	9	10	11
Index negativa	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

Las cadenas admiten indexación y el operador de corte **:** y son objetos **inmutables**.

Prueba en la Shell de Python:

Para la Shell	cad1[0]	cad1[-1]	cad1	cad1[:4]	cad1[-5:]
Resultado	H	s	Hola a todos	Hola	todos

>>> cad1[0]='h' da error!

TypeError: 'str' object does not support item assignment

MÁS TIPOS ESTRUCTURADOS: CADENAS II

>>> 'hola'+' a todos' proporciona 'hola a todos'

>>> '*'*5 proporciona '*****'

Revisa las posiciones del abecedario en el código ASCII y razona los resultados en la Shell de Python:

Para la Shell	'Gomez'<'Gonzalez'	'Gómez'<'Gonzalez'	'Martínez'<'martínez'	'Martín'<'Martínez'
Resultado	True	False	True	True

for caracter **in** cadena**:**

bloque

```
cadena='Hola a todos'
for i in range(len(cadena)):
    print(cadena[i])
```

```
for elemento in 'Hola a todos':
    print(elemento)
```

```
cadena='Hola a todos'
for elemento in cadena:
    print(elemento)
```

SECUENCIAS DE ESCAPE

Secuencias de escape. Son un par de caracteres, el primero \ y tienen un significado especial. Los dos caracteres denotan y ocupan el espacio de un único carácter. Por ejemplo, el salto de línea ocupa la posición 10 en el código Unicode.

\n Salto de línea

\t Tabulador horizontal

\v Tabulador vertical

\a Carácter de 'campana'

```
>>> print ('Hola\na todos')
```

Algunas funciones y métodos sobre str I

cadena=**str**(número)

La función predefinida str recibe un número y devuelve una cadena.

ord y chr. Son funciones predefinidas inversas.

número=**ord**(carácter)

ord recibe un carácter y devuelve su posición en el código Unicode.

caracter=**chr**(número)

chr recibe un número y devuelve el carácter que ocupa esa posición en el código Unicode.

cadena.**lower**() es un método que pasa a minúsculas cadena.

cadena.**upper**() es un método que pasa a mayúsculas cadena.

cadena.**title**() es un método que pasa a mayúsculas la inicial de cada palabra de la cadena de la izda.

Algunas funciones y métodos sobre str II

cadena.**replace**(patrón,reemplazo) es un método que busca patrón en cadena y lo reemplaza por reemplazo.

cadena.**split**(carácter_separador) es un método que recibe una cadena y la separa en trozos, usando el carácter_separador (por defecto, el espacio en blanco) volcando el resultado en una lista.

carácter_separador.**join**(lista_de_cadenas) es un método que recibe una lista_de_cadenas y usando el carácter_separador devuelve una cadena formada por los elementos de la lista con el carácter separador entre ellos.

Ejemplo. Prueba en la Shell:

>>> 'Estimados alumnos'.replace('o','@') obtiene Estimad@s alumn@s

>>> 'Hola a todos'.split() obtiene ['Hola', 'a', 'todos']

>>> ' '.join(['Hola', 'a', 'todos']) obtiene 'Hola a todos'

Más tipos estructurados: DICCIONARIOS I

¿Cómo almacenar en Python la información de alumnos y sus e-mails?

En principio, se pueden usar listas/tuplas, una para almacenar los nombres y otra para los e-mails, tal que cada pareja nombre, e-mail ocupe la misma posición en ambas listas/tuplas.

También se puede almacenar en una lista/tupla de listas/tuplas, una fila para cada alumno.

Pero, en ambos casos, el manejo de la información requiere usar índices y recorridos de las listas sin cesar, lo cual no es eficiente.

Más tipos estructurados: DICCIONARIOS II

Un **diccionario** es un tipo de datos estructurado **dict** que consiste en una sucesión de elementos desordenados del tipo clave: valor, con la restricción de que la clave debe ser inmutable (int, float, str, tuple).

Un diccionario se define por los símbolos llaves de apertura y cierre, que contienen a sus elementos separados por comas. Sea un diccionario de n elementos:

{clave_0 : valor_0, clave_1 : valor_1,…, clave_ n-1 : valor_n-1}

Ejemplo:

```
>>> dict1={'Luis Pérez': 'luispz@unican.es', 'José García':'josegr@unican.es'}
```

>>> type(dict1) devuelve dict

¿Cómo acceder al e-mail del primer elemento?

```
>>> dict1['Luis Pérez']
```

Más tipos estructurados: DICCIONARIOS III

Un diccionario es un objeto **mutable**, que admite modificaciones por indexación de cualquiera de sus elementos.

>>> dict1['Luis Pérez']='luisp1@unican.es' para cambiar el e-mail de este alumno.

Los valores de un diccionario pueden ser de cualquier tipo.

Para añadir un elemento a dict1:

dict1['nombre_alumno']='e-mail_alumno'

Para recorrer un diccionario:

for clave **in** dict1**:**

Para eliminar un elemento, trozo de un diccionario o el diccionario completo:

>>> **del** dict1['Luis Pérez'] elimina la pareja clave: valor del diccionario.

Algunas Funciones y Métodos para DICCIONARIOS

>>> **len**(dict1) devuelve el número de elementos del diccionario

Métodos útiles sobre diccionarios:

>>> dict1.**keys**() devuelve las claves de dict1

dict_keys(['Luis Pérez', 'José García'])

>>> dict1.**values**() devuelve los valores de dict1

dict_values(['luispz@unican.es', 'josegr@unican.es'])

>>> dict1.**items**() devuelve los elementos de dict1 (clave: valor)

dict_items([('Luis Pérez', 'luispz@unican.es'), ('José García', 'josegr@unican.es')])

Los tres métodos son iterables.

Para saber si una clave está en el diccionario:

>>> 'Luis Pérez' in dict1

Inicialización de DICCIONARIOS por comprensión (for implícito)

Sean dos listas L1 y L2 de la misma longitud, n.

Inicializar un diccionario con ambas listas de n elementos por comprensión :

dict1 = { L1[i] **:** L2[i] **for** i **in range(len(**L1**))}**

Ejemplo:

```
1 L1 = ['Luis Pérez','José García']
2 L2 = ['luispz@unican.es','josegr@unican.es']
3 dict1 = {L1[i]:L2[i] for i in range(len(L1))}
4 print(dict1)
```

En definitiva, una lista puede verse como un caso particular de un diccionario en el que la indexación se realiza con enteros.

Un diccionario es una estructura más general que una lista que se indexa con cualquier tipo inmutable.

Ejercicio 1a

Pedir por teclado el nombre y apellido de una persona. Hallar la longitud de la cadena nombre completo usando la función predefinida len() y sin usarla. Calcular cuantas 'aes' tiene, minúsculas o mayúsculas. Usa dos formas, una de ellas con indexación en la variable de tipo str, la otra sin indexación.

cap6_1a.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_1a.py (3.10.7)

File Edit Format Run Options Window Help

```
#PB Nov 2022
nom=input("Nombre ")
apel=input("Apellido ")
#concatenar nombre y apellido con espacio
nomc=nom+' '+apel
#print(nomc)
print("Longitud de ",nomc,": ",len(nomc))
conta=0
for caracter in nomc:
    conta+=1
print("Longitud de ",nomc,": ",len(nomc))
for letra in nomc:
    if letra=='a' or letra=='A':
        conta+=1
print("Cantidad de aes: ",conta)
#usando indexación
conta=0
for i in range(len(nomc)):
    if nomc[i]=='a' or nomc[i]=='A':
        conta+=1
print("Cantidad de aes: ",conta)
```

Ejercicio 1b

Construye una función que reciba una cadena y un carácter y calcule y devuelva la cantidad de apariciones del carácter en la cadena, independientemente de que esté en minúsculas o mayúsculas.

cap6_1b.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_1b.py (3.10.7)

File Edit Format Run Options Window Help

```
1 #PB Nov 2022
2 def ccd(cad,car):
3     conta=0
4     for ct in cad:
5         if ct==car.lower() or ct==car.upper():
6             conta+=1
7     return conta
```

>>>ccd('estA es la cadena','a')

4

Ejercicio 2

Escribir en pantalla una tabla con las posiciones del alfabeto español en minúsculas y mayúsculas en el código UNICODE. ¿Qué posición ocupa la ñ? ¿y la Ñ? ¿Hay alguna relación entre las posiciones de las letras minúsculas y mayúsculas?

cap6_2.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_2.py (3.10.7)

File Edit Format Run Options Window Help

```
1 #PB Nov 2022
2 alfabeto='abcdefghijklmnñopqrstuvwxyz'
3 for letra in alfabeto:
4     print(letra,ord(letra))
5 alfabetom=alfabeto.upper()
6 for letra in alfabetom:
7     print(letra,ord(letra))
```

Ejercicio 3

Construye una función para pasar a minúsculas un nombre que se recibe como argumento de entrada usando las funciones ord y chr. Suponer que el nombre leído puede tener mezcladas letras minúsculas y mayúsculas. No usar los métodos upper y lower. Tener en cuenta la ñ y Ñ.

cap6_3.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_3.py (3.10.7)

File Edit Format Run Options Window Help

```
1  #PB Nov 2022
2  def pasaminus(cade):
3      ''' pasa a minúsculas una palabra'''
4      cads=""
5      for caracter in cade:
6          nc=ord(caracter)
7          if 65<=nc<=90 or nc==209:
8              nc+=32
9              cads+=chr(nc)
10         else:
11             cads+=caracter
12     return cads
```

```
>>>pasaminus('COMpuTAcion')
```

Ejercicio 4

Construye una función para pasar a mayúsculas un nombre que se recibe como argumento de entrada usando las funciones ord y chr. Suponer que el nombre leído puede tener mezcladas letras minúsculas y mayúsculas. No usar los métodos upper y lower. Tener en cuenta la ñ y Ñ.

cap6_4.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_4.py (3.10.7)

File Edit Format Run Options Window Help

```
1  #PB Nov 2022
2  def pasamayus(cade):
3      ''' pasar a mayúsculas una palabra'''
4      cads=""
5      for caracter in cade:
6          nc=ord(caracter)
7          if 97<=nc<=122  or nc==241:
8              nc-=32
9              cads+=chr(nc)
10         else:
11             cads+=caracter
12     return cads
```

>>>pasamayus('FUNdamentos')

Ejercicio 5a

Construye una función para invertir una palabra. Dos formas de hacerlo, concatenando con el operador + por la izquierda de la cadena y por la derecha.

cap6_5a.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_5a.py (3.10.7)

File Edit Format Run Options Window Help

```
1  #PB Nov 2022
2  def invertir(cade):
3      ''' invertir una cadena'''
4      cadi=''
5      long=len(cade)
6  #concatenando por la izquierda
7      for letra in cade:
8          cadi=letra+cadi
9  ##concatenando por la derecha
10 ##     for i in range(long):
11 ##         cadi=cadi+cade[long-1-i]
12     return cadi
```

```
>>>invertir('Computación')
```

Ejercicio 5b

Usar el algoritmo siguiente: localizar la posición central de la palabra e intercambiar las posiciones de las letras última y primera, penúltima y segunda y así sucesivamente hasta llegar a la posición central de la palabra.

cap6_5b.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_5b.py (3.10.7)

File Edit Format Run Options Window Help

```
1  #PB Nov 2022
2  def invertir(cade):
3      ''' invertir una palabra'''
4      lcade=list(cade)
5      long=len(cade)
6      for i in range(long//2):
7          lcade[i],lcade[long-1-i]=lcade[long-1-i],lcade[i]
8  #usar join
9  #     cadi=''.join(lcade)
10     cadi=''
11     for elem in lcade:
12         cadi=cadi+elem
13     return cadi
```

```
>>>invertir('Computación')
```

Ejercicio 6

Construye una función con argumentos de entrada una sílaba y una palabra. Se trata de ver si está la sílaba en la palabra y en qué posición, empezando por la izquierda. No usar ningún método.

cap6_6.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_6.py (3.10.7)

File Edit Format Run Options Window Help

```
1 #PB Nov 2022
2 def buscasilaba(silaba, palabra):
3     if silaba in palabra:
4         for i in range(len(palabra)-len(silaba)):
5             if silaba==palabra[i:i+len(silaba)]:
6                 return True,i
7     else:
8         return False
```

```
>>> buscasilaba('tu','cacatua')
```

Ejercicio 7

Escribe un programa para buscar un nombre en una lista de nombres.

cap6_7a.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_7a.py (3.10.7)

File Edit Format Run Options Window Help

```
1  #PB Nov 2022
2  lnom=['Grace Hopper','Ada Lovelace','Heddy Lamarr','Angela Ruiz']
3  print('Lista de nombres: ', lnom)
4  nom=input('Dame un nombre: ')
5  #Forma 1. Usar operador de pertenencia in
6  if nom in lnom:
7      print("Encontrado")
8  if nom not in lnom:
9      print("El nombre no está en la lista")
10 #Forma 2. Sin método index
11 encontrado=False
12 i=0
13 while i<len(lnom):
14     if lnom[i]==nom:
15         encontrado=True
16         pos=i
17         print('Encontrado en posición: ',pos)
18     i+=1
19 if not encontrado:
20     print('El nombre no está en la lista')
```

Ejercicio 8

Ordenar alfabéticamente ascendente o descendentemente los nombres de varias personas. Utilizar el algoritmo de la burbuja y la función predefinida sorted().

cap6_8.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_8.py (3.10.7)

File Edit Format Run Options Window Help

```
#PB Nov 2022
def burbuja(lista):
    for i  in range(len(lista)-1,1,-1):
        for j in range(i):
            if lista[j]>lista[j+1]:
                lista[j],lista[j+1]=lista[j+1],lista[j]
def main():
    ln=['Ruiz Angela','Hopper Grace','Lovelace Ada','Lamarr Heddy']
    burbuja(ln)
    asc=int(input('¿Ordenación ascendente o descendente? 1/0 '))
    if asc:
        print('Lista ordenada ascendentemente:\n',ln)
        print('Idem: ',sorted(ln))
    else:
        lg=len(ln)
        lordd=[0]*lg
        lordd=[ln[lg-1-i] for i in range(lg)]
        print('Lista ordenada descendentemente:\n',lordd)
        print('Idem: ',sorted(ln,reverse=True))
```

```
>>> main()
```

Ejercicio 9a

Escribe un programa que pida por teclado indefinidamente un prefijo telefónico y escriba en pantalla el país correspondiente. Usar dos listas inicializadas en el programa, una con prefijos y otra con países. Si no está el prefijo en la lista de prefijos escribir un mensaje que lo indique.

cap6_9a.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\resueltos\cap6_9a.py (3.10.7)

File Edit Format Run Options Window Help

```
#PB Nov 2022
lpf=['34','33','32','49','39','31','46']
lps=['España','Francia','Bélgica', 'Alemania','Italia',
      'Paises Bajos','Suecia']
resp='s'
while resp.lower()=='s':
    pos=-1
    pf=input('Prefijo telefónico: ')
    for i in range(len(lpf)):
        if pf==lpf[i]:
            pos=i
            break
    if pos==-1:
        print('Ese prefijo no está en la lista')
    else:
        print('País: ',lps[pos])
    resp=input('¿Desea continuar (s/n)? ')
print("Gracias por usar este programa")
```

Ejercicio 9b

Usar un diccionario con claves los prefijos telefónicos y valores los países correspondientes. Si no está el prefijo en el diccionario, añadirlo.

cap6_9b.py - C:/DATOS/Curso 2022-2023/GIQ/LAB_PYTHON_22_23/cap6_Cadenas/resueltos/cap6_9b.py (3.10.7)

File Edit Format Run Options Window Help

```
1  #PB Nov 2022
2  lpf=['34','33','32','49','39','31','46']
3  lps=['España','Francia','Bélgica', 'Alemania','Italia',
4        'Paises Bajos','Suecia']
5  # Crea diccionario con las dos listas
6  d={lpf[i]:lps[i] for i in range(len(lpf))}
7  print('Diccionario de prefijos telefónicos:\n',d)
8  resp='s'
9  while resp.lower()=='s':
10     pf=input('Prefijo telefónico: ')
11     if pf in d:
12         print('País: ',d[pf])
13     else:
14         d[pf]=input("País ")
15         print("Añadido elemento a diccionario")
16     resp=input('¿Desea continuar (s/n)? ')
17 print("Gracias por usar este programa")
```

Ejercicio 10 I

Tomando como base el ejercicio 11 propuesto del capítulo 5, construir un programa con 7 procedimientos. Cada procedimiento dibuja una letra de las palabras FELICES FIESTAS. Cada procedimiento se llama dibuja_X (con X=F, E, L, I, C, S, T, A) y construye una matriz cuadrada recibiendo dos argumentos de entrada: el símbolo usado para dibujar y el tamaño de la matriz, el cual debe ser impar y por defecto 11.

Completa el programa siguiente añadiendo los procedimientos para las letras F, L y T.

Dedicado a los alumn@s del curso 2021-22 que colaboraron en ello.

Ejercicio 10 II

dibuja1.py - C:\DATOS\Curso 2022-2023\GIQ\LAB_PYTHON_22_23\cap6_Cadenas\ActividadLetra(Colaborativo)\Solución\dibuja1.py (3.10.7)

File Edit Format Run Options Window Help

```
#PB Nov 2022
def vermat(m):    #procedimiento
    '''escribe matriz en pantalla, sólo elementos'''
    for i in range(len(m)):
        for j in range(len(m[0])):
                        print(m[i][j],end=' ')
        print()
    print()

def dibuja_E(s,t=11):
    '''dibuja E con matriz cuadrada txt usando el símbolo s '''
    matriz=[[' ']*t]*t
    matriz[0]=[s]*t #primera fila
    matriz[t//2]=[s]*t #fila del medio
    matriz[t-1]=[s]*t #última fila
    for fila in matriz:
        fila[0]=s       #primera columna
    vermat(matriz)

def dibuja_I(s,t=11):
    '''dibuja I con matriz cuadrada txt usando el símbolo s '''
    matriz=[[' ' for i in range(t)] for j in range(t)]
    for fila in matriz:
        fila[t//2]=s
    vermat(matriz)

def dibuja_S(s,t=11):
    '''dibuja S con matriz cuadrada txt usando el símbolo s '''
    matriz=[[' ' for i in range(t)] for j in range(t)]
    matriz[0]=[s]*t
    matriz[t//2]=[s]*t
    matriz[t-1]=[s]*t
    for fila in range(t):
        if fila < t//2:
            matriz[fila][0]=s
        else:
            matriz[fila][t-1]=s
    vermat(matriz)
```

Ejercicio 10 III

```
65 def dibuja_C(s,t=11):
66     '''dibuja C con matriz cuadrada txt usando el símbolo s '''
67     matriz=[[' ']*t]*t
68     matriz[0]=[s]*t #primera fila
69     matriz[t-1]=[s]*t #última fila
70     for fila in matriz:
71         fila[0]=s       #primera columna
72     vermat(matriz)
73
74 def dibuja_A(s,t=11):
75     '''dibuja A con matriz cuadrada txt usando el símbolo s '''
76     matriz=[[' ']*t]*t
77     matriz[0]=[s]*t #primera fila
78     matriz[t//2]=[s]*t #fila del medio
79     for fila in matriz:
80         fila[0]=s       #primera columna
81         fila[t-1]=s     #última columna
82     vermat(matriz)
83
84 def dibuja__(s,t=11):
85     '''dibuja un espacio blanco con matriz cuadrada txt usando el símbolo s '''
86     matriz=[[' ']*t]*t
87     vermat(matriz)
88
89 print("¿Preparado para leer un mensaje especial?")
90 pausa=input("Si estás preparado, pulsa una tecla")
91 print()
92 t=7 #tamaño de letra, mejor impar
93 ##eval('dibuja_{0}({1},{2})'.format('F',"'f'",t))
94 for letra in "FELICES_FIESTAS":
95     eval('dibuja_{0}("{1}",{2})'.format(str(letra),str(letra),t))
96 print((chr(0x26c4)+'  ')*2," F E L I C E S ",(chr(0x26c4)+'  ')*2)
97 print(('  '+chr(0x2744))*2," F I E S T A S ",('  '+chr(0x2744))*2)
98 print()
```

Ejercicio 10 IV Ejecución