Activitats de la Sessió 3

2. Mètodes de les cadenes o strings

capitalize()
Retorna otra cadena con el primer caracter en mayúsculas.

```
>>> cadena='ana'
>>> print cadena.capitalize()
Ana
>>> print cadena
ana
>>>
```

upper()
Retorna otra cadena con todos los caracteres convertidos a mayúscula.

```
>>> cadena1='ana'
>>> cadena2=cadena1.upper()
>>> print cadena2
ANA
>>>
```

lower()
Retorna otra cadena con todos los caracteres convertidos a minúsculas.

```
>>> cadena1='Ana'
>>> cadena2=cadena1.lower()
>>> print cadena2
ana
```

isupper()
Retorna True si todos los caracteres de la cadena están en mayúsculas.

```
>>> cadena='ANA'
>>> if cadena.isupper():
... print 'La cadena ',cadena,' esta toda en mayusculas'
La cadena  ANA  esta toda en mayusculas
```

```
#islower()

cadena1='ana'
if cadena.islower():
    print 'La cadena '+cadena+ ' esta toda en minusculas'

#isdigit()
cadena='120'
if cadena.isdigit():
    print 'Todos los caracteres de la cadena son números'

#isalpha()
cadena='Hola mundo'
if cadena.isalpha():
    print 'Todos los caracteres de la cadena son del alfaberto'
else:
    print 'No todos los caracteres de la cadena son del alfabeto'

#isspace()
cadena= '  '
if cadena.isspace():
    print 'Todos los caracteres de la cadena son espacios en blanco'
```

```
#isalnum()
cadena='cordoba2008'
if cadena.isalnum():
    print 'Todos los caracteres son numeros o alfabeticos'

#find('cadena',[inicio],[fin])
cadena='esto es una prueba y es solo eso'
pos=cadena.find('es')
print pos

cadena='esto es una prueba y es solo eso'
pos=cadena.find('es',5)
print pos

#rfind('cadena',[inicio],[fin])
cadena='esto es una prueba y es solo eso'
pos=cadena.rfind('es')
print pos

#count('cadena',[inicio],[fin])
cadena='esto es una prueba y es solo eso'
cant=cadena.count('es')
print cant
```

```
#replace('cadena1','cadena2',[maximo])
cadena1='esto es una prueba y es solo eso'
cadena2=cadena1.replace('es','ES')
print cadena2

#split('caracter separador',[maximo])
cadena='esto es una prueba y es solo eso'
lista=cadena.split(' ')
print lista
print len(lista)
lista=cadena.split(' ',2)
print lista
print len(lista)

#rsplit('caracter separador',[maximo])
cadena='esto es una prueba y es solo eso'
lista=cadena.rsplit(' ')
print lista
print len(lista)
```

Resultados:

```
sergio@sergio-desktop:~/Escritorio$ python ejemplos.py
La cadena ana esta toda en minusculas
Todos los caracteres de la cadena son numeros
No todos los caracteres de la cadena son del alfabeto
Todos los caracteres de la cadena son espacios en blanco
Todos los caracteres son numeros o alfabeticos
0
5
29
4
ESto ES una prueba y ES solo ESo
['esto', 'es', 'una', 'prueba', 'y', 'es', 'solo', 'eso']
8
['esto', 'es', 'una prueba y es solo eso']
3
['esto', 'es', 'una', 'prueba', 'y', 'es', 'solo', 'eso']
8
['esto es una prueba y es', 'solo', 'eso']
3
['Primer linea', 'Segunda linea', 'Tercer linea', 'Cuarta linea']
sISTEMA DE fACTURACION
$$200
200$$
$200$
```

3. Mètodes de les llistes

```
#append(elemento)
lista=['juan','ana','luis']
lista.append('carlos')
print lista

#extend(elementos)
lista=['juan','ana','luis']
lista.extend(['uno','dos'])
print lista

lista=['juan','ana','luis']
lista.append(['uno','dos'])
print lista

#insert(posición,elemento)
lista=['juan','ana','luis']
lista.insert(0,'carlos')
print lista
#pop([posicion])
lista=['juan','ana','luis','marcos']
elemento=lista.pop()
print elemento
print lista
print lista.pop(1)
print lista

#remove(elemento)
lista=['juan','ana','luis','marcos','ana']
lista.remove('ana')
print lista

#count(elemento)
lista=['juan','ana','luis','marcos','ana']
print lista.count('ana')

#index(elemento,[inicio],[fin])
lista=['juan','ana','luis','marcos','ana']
print lista.index('ana')

#sort()
lista=['juan','ana','luis','marcos','ana']
lista.sort()
print lista
```

```
#sort()
lista=['juan','ana','luis','marcos','ana']
lista.sort()
print lista

#reverse()
lista=['juan','ana','luis','marcos','ana']
lista.reverse()
print lista

#Borrado de elementos de la lista
lista=['juan','ana','luis','marcos']
del lista[2]
print lista
#Si queremos borrar los elementos de la posición 2 hasta la 3:
lista=['juan','ana','carlos','maria','pedro']
del lista[2:4]
print lista
#Si queremos borrar desde la 2 hasta el final:
lista=['juan','ana','carlos','maria','pedro']
del lista[2:]
print lista
#Si queremos borrar todos desde el principio hasta la posición 3 sin incluirla:
lista=['juan','ana','carlos','maria','pedro']
del lista[:3]
print lista
#Si queremos ir borrando de a uno de por medio:
lista=['juan','ana','carlos','maria','pedro']
del lista[::2]
print lista
#Si necesitamos modificar el contenido de un nodo de la lista
lista=['juan','ana','luis','marcos']
lista[2]='xxxxx'
print lista

#Conocer la cantidad de elementos actual
#Se utiliza la función len():
lista=['juan','ana','luis','marcos']
print len(lista)
```

Resultados:

```
sergio@sergio-desktop:~/Escritorio$ python ejemplos2.py
['juan', 'ana', 'luis', 'carlos']
['juan', 'ana', 'luis', 'uno', 'dos']
['juan', 'ana', 'luis', ['uno', 'dos']]
['carlos', 'juan', 'ana', 'luis']
marcos
['juan', 'ana', 'luis']
ana
['juan', 'luis']
['juan', 'luis', 'marcos', 'ana']
2
1
['ana', 'ana', 'juan', 'luis', 'marcos']
['ana', 'marcos', 'luis', 'ana', 'juan']
['juan', 'ana', 'marcos']
['juan', 'ana', 'pedro']
['juan', 'ana']
['maria', 'pedro']
['ana', 'maria']
['juan', 'ana', 'xxxxx', 'marcos']
4
```

4. Mètodes dels diccionaris

```
#keys()
diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
lista=diccionario.keys()
print lista

#values()
diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
lista=diccionario.values()
print lista

#items()
diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
lista=diccionario.items()
print lista

#pop(clave,[valor])
diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
valor=diccionario.pop('window')
print valor
print diccionario

diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
valor=diccionario.pop('love','clave no encontrada')
print valor
```

```
#has_key(clave)
diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
if diccionario.has_key('love'):
print 'Si tiene la clave buscada'
else:
print 'No existe la clave buscada'

#clear()
diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
diccionario.clear()
print diccionario

#copy()
diccionario1={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
diccionario2=diccionario1.copy()
print diccionario2
diccionario1['house']='xxxxx'
print diccionario2

#popitem()
diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
elemento=diccionario.popitem()
print elemento
print diccionario
```

```
#update(diccionario2)
diccionario1={'uno':'1','dos':'2','tres':'3333'}
diccionario2={'tres':'3','cuatro':'4','cinco':'5'}
diccionario1.update(diccionario2)
print diccionario1

#Borrado de elementos del diccionario
diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
del diccionario['house']
print diccionario
#Modificación y creación de elementos del diccionario
diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
diccionario['red']='colorado'
diccionario['blue']='azul'
print diccionario
#Conocer la cantidad de elementos actual
diccionario={'house':'casa','red':'rojo','bed':'cama','window':'ventana'}
print len(diccionario)
```

Resultados:

```
sergio@sergio-desktop:~/Escritorio$ python ejemplos3.py
['house', 'window', 'bed', 'red']
['casa', 'ventana', 'cama', 'rojo']
[('house', 'casa'), ('window', 'ventana'), ('bed', 'cama'), ('red', 'rojo')]
ventana
{'house': 'casa', 'bed': 'cama', 'red': 'rojo'}
clave no encontrada
No existe la clave buscada
{}
{'house': 'casa', 'window': 'ventana', 'red': 'rojo', 'bed': 'cama'}
{'house': 'casa', 'window': 'ventana', 'red': 'rojo', 'bed': 'cama'}
('house', 'casa')
{'window': 'ventana', 'bed': 'cama', 'red': 'rojo'}
{'cuatro': '4', 'cinco': '5', 'dos': '2', 'tres': '3', 'uno': '1'}
{'window': 'ventana', 'bed': 'cama', 'red': 'rojo'}
{'blue': 'azul', 'house': 'casa', 'window': 'ventana', 'bed': 'cama', 'red': 'colorado'}
4
```

4. Activitats de la Sessió 3

1. Llistes

Crear un fitxer d'script amb el següent contingut d'exemple:

```
sergio@sergio-desktop:~/Escritorio/Introducción Python/03- ejercicios python$ py
thon ex1.py
[105, 12, 145, 62, 116, 111, 106, 18, 254, 197, 194, 26, 97, 62, 123, 278, 94, 1
18, 189, 236, 262, 118, 107, 1, 205, 107, 218, 161, 248, 188, 251, 29, 103, 41,
86, 198, 160, 192, 213, 117, 103, 197, 224, 45, 214, 37, 251, 34, 247]


[12, 145, 62, 116, 111, 106, 18, 254, 197, 194, 26, 97, 62, 123, 278, 94, 118, 1
89, 236, 262, 118, 107, 1, 205, 107, 218, 161, 248, 188, 251, 29, 103, 41, 86, 1
98, 160, 192, 213, 117, 103, 197, 224, 45, 214, 37, 251, 34]


[12, 145, 62, 116, 111, 106, 18, 254, 197, 194, 26, 97, 62, 123, 278, 94, 118, 1
89, 236, 262, 118, 107, 1, 205, 107, 218, 161, 248, 188, 251, 29, 103, 41, 86, 1
98, 160, 192, 213, 117, 103, 197, 224, 45, 214, 37, 251, 34, 6536]


[12, 125, 145, 62, 116, 111, 106, 18, 254, 197, 194, 26, 97, 62, 123, 278, 94, 1
18, 189, 236, 262, 118, 107, 1, 205, 107, 218, 161, 248, 188, 251, 29, 103, 41,
86, 198, 160, 192, 213, 117, 103, 197, 224, 45, 214, 37, 251, 34, 6536]
sergio@sergio-desktop:~/Escritorio/Introducción Python/03- ejercicios python$
```

```python
import random

#Crear una lista con 30 valores aleatorios comprendidos entre 1 y 300

lista = []

for x in range(1,50):
    valor = random.randint(1,300)
    lista.append(valor)
print lista
print '\n'

#Borrar el primer y ultimo elemento de la lista
del lista[0]
del lista[-1]
print lista
print '\n'

#Insertar un elemento al final con la suma de todos los elementos actuales
suma=0
for x in range(1,len(lista)):
    suma=suma+lista[x]
lista.append(suma)
print lista
print '\n'

#Insertar un elemento entre el primero y el segundo elemento de la lista con el valor 125.
lista.insert(1,125)
print lista
```

Executar-lo i analitzar com funciona.

Programar una variació del programa que

- Crei una llista amb 10 valors aleatoris entre 1 i 5.
- Inserti un element al final amb el màxim valor de la llista
- Inserti un element al principi, amb la suma dels primers 5 elements
- Imprimeixi les vegades que apareix a la llista el valor de segon element

Resultado:

```
sergio@sergio-desktop:~/Escritorio$ python ex1.1.py
llista amb 10 valors aleatoris entre 1 i 5
[4, 2, 2, 2, 5, 4, 3, 5, 5, 5]
Insertar al final el maxim valor de la llista
[4, 2, 2, 2, 5, 4, 3, 5, 5, 5, 5]
Insertar ual principi amb la suma dels primers 5
[15, 4, 2, 2, 2, 5, 4, 3, 5, 5, 5, 5]
('les vegades que apareix a la llista el valor de segon element', 6)
```

```
import random

#crei una llista amb 10 valors aleatoris entre 1 i 5.
lista = []
for x in range(1,11):
    valor = random.randint(1,5)
    lista.append(valor)
print "llista amb 10 valors aleatoris entre 1 i 5"
print lista

#inserti un element al final amb el maxim valor de la llista
lista.insert(11,5)
print "Insertar al final el maxim valor de la llista"
print lista

#inserti un element al principi, amb la suma dels primers 5 elements
suma=0
print "Insertar ual principi amb la suma dels primers 5"
for x in range(0,5):
    suma=suma+lista[x]
lista.insert(0,suma)
print lista

#imprimeixi les vegades que apareix a la llista el valor de segon element
veces = 0
for i in lista:
   lista.count(lista[2])
   i=i+1
print("les vegades que apareix a la llista el valor de segon element", i)
```

2. Diccionaris.

```
#asigna un valor a las frutas
frutas = {'manzanas':1.60,'peras':1.90,'bananas':0.95}
print frutas
print '\n'

#añade una fruta mas. len se usa para obtener la longitud,
#es decir, para conocer la cantidad de frutas que hay
frutas['naranjas']=2.50
print '\n'

#del borra 'naranjas'. El bucle mediante keys() devuelve la lista con
#todas las frutas menos con naranjas
del frutas['naranjas']
for x in frutas.keys():
    print x
print '\n'

#Escribe el valor de las frutas
for x in frutas.values():
    print x
print '\n'

#El bucle recorre las clave valor de la lista de frutas y las devuelve
#Se verán por pantalla una lista de claves/valor
for (clave,valor) in frutas.items():
    print clave+' '+str(valor)+'-'
print '\n'

#Borra el diccionario de frutas
frutas.clear()
print frutas
```

Resultado:

```
sergio@sergio-desktop:~/Escritorio$ python ex2.py
{'peras': 1.9, 'bananas': 0.95, 'manzanas': 1.6}



peras
bananas
manzanas


1.9
0.95
1.6


peras 1.9-
bananas 0.95-
manzanas 1.6-


{}
```

3. Llistes.

```
ten_things = "Apples Oranges Crows Telephone Light Sugar"
print "Wait there are not 10 things in that list. Let's fix that"

stuff = ten_things.split(' ')
more_stuff=["Day","Night","Song","Frisbee","Corn","Banana","Girl","Boy"]

while len(stuff) !=10:
    netx_one=more.stuff.pop()
    print "Adding: ",next_one
    stuff.append(next_one)
    print "There are %d items now" % len(stuff)

print "There we go: ",stuff
print "Let's do some things with stuff"

print stuff[1]
print stuff[-1]
print stuff.pop()
print ' '-join(stuff)
print '#'-join(stuff[3:5])
```

```
sergio@sergio-desktop:~/Escritorio$ python ex3.py
Wait there are not 10 things in that list. Let's fix that
Adding:  Boy
There are 7 items now
Adding:  Girl
There are 8 items now
Adding:  Banana
There are 9 items now
Adding:  Corn
There are 10 items now
There we go:  ['Apples', 'Oranges', 'Crows', 'Telephone', 'Light', 'Sugar', 'Boy',
'Girl', 'Banana', 'Corn']
Let's do some things with stuff
Oranges
Corn
Corn
Apples Oranges Crows Telephone Light Sugar Boy Girl Banana
Telephone#Light
```

Executar-lo i comprendre com funciona el programa.
Comentar l'efecte dels mètodes emprats.
Descriure l'efecte del mètode "join".
(Veure http://www.tutorialspoint.com/python/string_join.htm)

· Se crea una variable, ten_things, que contendrá una frase. Más adelante, vemos como esta frase que contiene la variable se va a separar mediante espacios, cogiendo de esta manera palabra por palabra toda la frase. More_stuff contiene una lista de palabras. Nuestro bucle while recorrerá la lista (la cortada por splits) y tomará la última palabra (recordemos que pop elimina). Cuando va a eliminar una palabra, la printa.

Los siguientes prints imprimen los elementos que están en la posición especificada. Join concatenará mediante un espacio de por medio, todas las palabras

4. Diccionaris.

```
states={
    'Oregon': 'OR',
    'Florida': 'FL',
    'California': 'CA',
    'New York': 'NY',
    'Michigan': 'MI',
}

cities={
    'CA': 'San Francisco',
    'MI': 'Detroit',
    'FL': 'Jacksonville',
}
#add some more cities
cities['NY'] = 'New York'
cities['OR'] = 'Portland'
#print out some cities
print '-' *10
print "NY State has: ",cities['NY']
print "OR State has: ",cities['OR']
#print some states
print '-'*10
print "Michigan's abbreviation is: ",states['Michigan']
print "Florida's abreviattion is: ",states['Florida']
```

```
#do it by using the state then cities dict
print '-' * 10
print "Michigan has: ",cities[states['Michigan']]
print "Florida has: ",cities[states['Florida']]
#print every state abbreviation
print '-'*10
for state, abbrev in states.items():
    print "%s is abbreviated %s" %(state,abbrev)
#print every city in state
print '-'*10
for abbrev,city in cities.items():
    print "%s has the city %s" %(state,abbrev)
#now do both at the same time
print '-' *10
for state, abbrev in states.items():
    print "%s state is abbreviated %s and has city %s" % (state,abbrev,cities[abbrev])
print '-'*10
#safely get a abbreviation by state that might not be there
state=states.get('Texas')

if not state: #Tots els objectes es poden avaluar com un booleà.
    print "Sorry, no Texas"

#get a city with a default value
city=cities.get('TX','Does Not Exist')
print "The city for the state 'TX' is: %s" %city
```

```
sergio@sergio-desktop:~/Escritorio$ python ex4.py
----------
NY State has:  New York
OR State has:  Portland
----------
Michigan's abbreviation is:  MI
Florida's abreviattion is:  FL
----------
Michigan has:  Detroit
Florida has:  Jacksonville
----------
California is abbreviated CA
Michigan is abbreviated MI
New York is abbreviated NY
Florida is abbreviated FL
Oregon is abbreviated OR
----------
Oregon has the city FL
Oregon has the city CA
Oregon has the city MI
Oregon has the city OR
Oregon has the city NY
----------
California state is abbreviated CA and has city San Francisco
Michigan state is abbreviated MI and has city Detroit
New York state is abbreviated NY and has city New York
Florida state is abbreviated FL and has city Jacksonville
Oregon state is abbreviated OR and has city Portland
----------
Sorry, no Texas
The city for the state 'TX' is: Does Not Exist
sergio@sergio-desktop:~/Escritorio$
```

Executar-lo i comprendre com funciona el programa.
Comentar l'efecte dels mètodes emprats.
Se crean dos diccionarios, states y cities y en ellos se van añadiendo datos sobre ciudades y estados. Se usan diversos métodos para añadir elementos al diccionario. Además tenemos bucles que printa las abreviaciones, estados y ciudades. Al final del código tenemos creado un “valor por defecto” en el que Texas siempre saldrá como no existente

Prova d'assignar una ciutat a un estat que ja en té una d'assignada. Què passa?

```
states={
        'Oregon': 'OR',
        'Florida': 'FL',
        'California': 'CA',
        'New York': 'NY',
        'Michigan': 'MI',
    }

cities={
        'CA': 'San Francisco',
        'MI': 'Detroit',
        'FL': 'Jacksonville',
        'CA' : 'Cadiz',
    }
```

```
California state is abbreviated CA and has city Cadiz
Michigan state is abbreviated MI and has city Detroit
New York state is abbreviated NY and has city New York
Florida state is abbreviated FL and has city Jacksonville
Oregon state is abbreviated OR and has city Portland
```

La ciudad añadida es CA, Cadiz

5. Diccionaris.
Adaptar el codi de l'activitat anterior, de manera que es puguin guardar diverses ciutats per cada estat.