In [2]:
```
# Ejemplo
usuarios = { 'Marta', 'David', 'Elvira', 'Juan' , 'Marcos' }
print(type (usuarios))

administradores  = { 'Marta',  'Juan'  }

administradores -= {'Juan'}
print (administradores)
```

```
<class 'set'>
{'Marta'}
```

In [5]:
```
# test_conjuntos.py

c1 = set ()                      # Un nuevo conjunto vacío
c1.add ("gato")                          # Agregar un solo elemento
c1.update (["perro", "ratón"])  # Agregar varios elementos,
                                 # por extensión de la lista
c1 |= set (["asno", "caballo"])          # Agregar 2 elementos
if "gato" in c1 :                       # Esta en la colección ?
  c1.remove ("gato")
# c1.remove ("elefante") arroja un error
c1.discard ("elefante")                  # No lanza ningún error
print (c1)

for item in c1:              # Iteracion for each element
  print (item)
print ("Item count:", len(c1)) # Longitud /tamaño/núm.elementos primero = c1[0]
            # Error: no  hay índices para conjunt
es_vacio = len(c1) == 0      # Prueba de vacío

c1 = {"gato", "perro"}           # Inicializar el set usando llaves;
#c1 = {}                     # Esto no es un diccionario
c1 = set(["gato", "perro"])     # Inicializar  usando una lista
c2 = set(["perro", "raton"])
c3 = c1 & c2                     # Intersección
```

```
{'asno', 'caballo', 'ratón', 'perro'}
asno
caballo
ratón
perro
Item count: 4
```

In [7]:
```
c4 = c1 | c2              # Union
c5 = c1 - c3              # Diferéncia de conjuntos
c6 = c1 ^ c2              # Diferéncia simétrica
issubc = c1 <= c2         # Prueba de subconjunto
issuperc = c1 >= c2       # Prueba de superconjunto c7 =

c7 = c1.copy()                  # Crea una copa
c7.remove("gato")
print (c7.pop())              # Elimina un elemento arbitrario

c8 = c1.copy()
c8.clear()                   # Limpia, vacia el conjunto
c9 = {x for x in range(10) if x % 2}

# c comprehension; since Python 2.7
print (c1, c2, c3, c4, c5, c6, c7, c8, c9, issubc, issuperc)
```

```
perro
{'perro', 'gato'} {'perro', 'raton'} {'perro'} {'perro', 'gato', 'raton'} {'gato'}
{'gato', 'raton'} set() set() {1, 3, 5, 7, 9} False False
```

In [46]:
```
#PRACTICA P01. Ejercicio 1
#-------------------------------------------------------
# Sea el conjunto c1 = {1, 2, 3, 4}
# Añade el 5, descarta el 2 usando métodos.
# Agrega 3 números aleatorios (random) del 1 al 20  (*)
# Crea otro conjunto llamado c2 con los cuadrados del conjunto c1.
# Calcula e imprime la unión, intersección y diferencia de los conjuntos c1 y c2.
from random import sample, randrange

c1 = {1, 2, 3, 4}
c1.add(5)
c1.discard (2)
print(c1)

# https://pynative.com/python-random-randrange/
c1.update ( {x for x in sample(range(1, 21), 3)} )
print("c1: ", c1)

c2 = { x**2 for x in c1 }
print("c2: ",c2)

print("c1 | c2: ", c1.union(c2))
print("c1 & c2: ", c1.intersection(c2))
print("c1 - c2: ", c1.difference(c2))
```

```
{1, 3, 4, 5}
c1:  {1, 3, 4, 5, 12, 17, 19}
c2:  {1, 289, 9, 361, 16, 144, 25}
c1 | c2:  {1, 289, 3, 4, 5, 9, 361, 12, 16, 17, 144, 19, 25}
c1 & c2:  {1}
c1 - c2:  {3, 4, 5, 12, 17, 19}
```

In [7]:
```
from random import randint
c8 =set()

# for x in range(3):
#     c8.add(random.randint(21))

while len(c8) < 3:
    c8.add(randint(0,21))

print(c8)
```

```
{17, 11, 6}
```

In [11]:
```
from random import sample, randrange
c1 = {1, 2, 3, 4}
c1.update ( {x for x in sample(range(1, 21), 3)} )
print (c1)
```

```
{1, 2, 3, 4, 8, 11, 13}
```

In [20]:
```
#PRACTICA P01. Ejercicio 2
#-----------------------------------------------------
# Funcion modificar que apartir de una lista de números
#   realice las siguientes acciones sin variar la original
#  - Borrar los elementos duplicados
#  - Ordenar la lista de mayor a menor
#  - Eliminar todos los números impares
#  - Realizar una suma de todos los números que quedan
#  - Añadir como primer elemento de la lista la suma realizada
#  - Devolver la lista modificada
#  - Finalmente, despues de ejecutar la función, comprueba que la suma de todos
#    los números a partir del segundo concuerda con el primer número de la lista.
#-------------------------------------------------------------------------------

lista = [1,2,2,3,3,4,4]
lista = list (set(lista))
lista.sort(reverse=True)

lista = [x for x in lista if x %2 != 0]

lista.insert(0, sum(lista))

print(lista)


print (lista[0] == sum (lista[1:]))
```

```
[4, 3, 1]
True
```

In [2]:
```
#PRACTICA P01. Ejercicio 3

# lista1 = input ("Entri conjunt 1 (separat per comes) : ")
# lista1 = lista1.replace(" ", "").split(",")

# lista2 = input ("Entri conjunt 2 (separat per comes) : ")
# lista2 = lista2.replace(" ", "").split(",")
```

In [4]:
```
# PRACTICA P02
cadena = 'T7 Prom 5-d CGC CAG GGT TTT TCA AGT CAC GAC GAC AGT TAT TGC TCA GCG G '
```

In [25]:
```
#P03--------------------------------------------------------------------------
cadena = "La experiencia de aprendizaje de Python difiere bastante en función de la
Sea cual sea el lenguaje informático practicado, es necesario tener cierta lógica y
conceptos algorítmicos. \
Escoger Python como primer lenguaje es la mejor elección que puede realizar: muy pr
los conceptos algorítmicos clásicos, le permitirá hacer gran cantidad de cosas de m
una curva de aprendizaje muy pronunciada. \
Esta experiencia de aprendizaje difiere bastante según los lenguajes practicados en
lenguaje aporta su propia manera de pensar y su implementación de las técnicas algo
pensamiento del que lo practica."


cadena = cadena.lower().replace(".", "").replace(",", "")

palabras = cadena.split(" ")            # Convierte el texto en una lista
print ("hay : ",len(palabras))        # Cuantas plabras hay
c = set(palabras)                       # convierte lista en conjunto

cexcluye = {"el", "la", "los", "las", "un", "de"} #conjunto de palabras a eliminar
c = c - cexcluye
distintas  = list(c)              # y pasa el conjunto a otra lista palabras pero y
print ("hay distintas:",len(distintas))          # Cuantas palabras diferentes hay


distintas.sort()                # Ordena la lista

print ("{:15} {:5} ".format("palabras", "veces"))
print ("{:15} {:5} ".format("--------", "-----"))

palab_mas_5 = []   # Lista de palabras de más de 5 letras

for palabra in distintas :         # para cada palabra diferente ....

    if len(palabra) > 5 :
        palab_mas_5.append( ( palabras.count(palabra), palabra )   )
                                    # cuenta cuantas veces aparecia en la lista or

        print ("{:15} {:5}".format(palabra, palabras.count(palabra)) )

print ("palabras de más de 5 letras  : ", len(palab_mas_5))

# ------- El print sigiuiente sale ordenado por frecuencia
palab_mas_5.sort(reverse=True)
print(palab_mas_5)
```

```
hay :  117
hay distintas: 69
palabras        veces
--------        -----
algorítmicas        1
algorítmicos        2
aporta              1
aprendizaje         3
aprovechar          1
bastante            2
cantidad            1
cierta              1
ciertos             1
clásicos            1
conceptos           2
difiere             2
dominar             1
efecto              1
elección            1
escoger             1
experiencia         3
función             1
implementación      1
informático         1
lenguaje            4
lenguajes           1
lógica              1
manera              2
moldea              1
natural             2
necesario           1
pasado              1
pensamiento         1
pensar              1
permitirá           1
practica            1
practicado          1
practicados         1
primer              1
pronunciada         1
propia              1
próximo             1
python              2
realizar:           1
técnicas            1
palabras de más de 5 letras  :  41
[(4, 'lenguaje'), (3, 'experiencia'), (3, 'aprendizaje'), (2, 'python'), (2, 'natur
al'), (2, 'manera'), (2, 'difiere'), (2, 'conceptos'), (2, 'bastante'), (2, 'algorí
tmicos'), (1, 'técnicas'), (1, 'realizar:'), (1, 'próximo'), (1, 'propia'), (1, 'pr
onunciada'), (1, 'primer'), (1, 'practicados'), (1, 'practicado'), (1, 'practica'),
(1, 'permitirá'), (1, 'pensar'), (1, 'pensamiento'), (1, 'pasado'), (1, 'necesario
'), (1, 'moldea'), (1, 'lógica'), (1, 'lenguajes'), (1, 'informático'), (1, 'implem
entación'), (1, 'función'), (1, 'escoger'), (1, 'elección'), (1, 'efecto'), (1, 'do
minar'), (1, 'clásicos'), (1, 'ciertos'), (1, 'cierta'), (1, 'cantidad'), (1, 'apro
vechar'), (1, 'aporta'), (1, 'algorítmicas')]
```

In [63]:
```
import re
text = 'My name is Abder'
res = re.findall('Merce', text)
print (res)
```

```
[]
```

In [75]:
```
import re

#---------------------------------------------------------------------
# Localiza en un texto la frecuencia de las palabras de más de 6 letras
#---------------------------------------------------------------------

veces = {}    # crea un diccionario vacio para guardar las veces de cada palabra
fiche = open('dat\\texto2.txt', 'r', encoding='utf-8')
texto = fiche.read().lower()
print("texto base", texto)

# obtiene una lista de las palabras de 6 a 15 letras
lista_palabras = re.findall(r'[a-z]{6,15}', texto, re.MULTILINE)

print ("Total palabras de más de 6 letras: ", len(lista_palabras))

for p in lista_palabras:
    count = veces.get(p,0)
    veces[p] = count + 1

claves = veces.keys()

for p in claves:
    print ( p, ": ", veces[p])
```

```
texto base un tutorial de python diferente
este curso de python es diferente a los demás. la mayoría de los cursos, intentan e
xplicar todas las características del lenguaje de principio a fin. esto, además de
ser arduo, hace que muchas personas se frustren ante semejante avalancha de informa
ción, y tiren la toalla antes de tiempo.

en vez de pasar por todas y cada una de las posibilidades que ofrece python, que si
n lugar a dudas puede ser abrumador, en este curso aprenderás lo básico para

la intención de este curso de python es que aprendas a programar lo más rápido posi
ble, por lo que en vez de desmenuzar todas las características, se centra en lo mín
imo indispensable para desenvolverse con soltura. está diseñado para personas sin e
xperiencia en programación, aunque si ya sabes algo, aprenderás aún más rápido
Total palabras de más de 6 letras:  42
tutorial :  1
python :  4
diferente :  2
cursos :  1
intentan :  1
explicar :  1
caracter :  2
sticas :  2
lenguaje :  1
principio :  1
muchas :  1
personas :  2
frustren :  1
semejante :  1
avalancha :  1
informaci :  1
toalla :  1
tiempo :  1
posibilidades :  1
ofrece :  1
abrumador :  1
aprender :  2
intenci :  1
aprendas :  1
programar :  1
posible :  1
desmenuzar :  1
centra :  1
indispensable :  1
desenvolverse :  1
soltura :  1
experiencia :  1
programaci :  1
aunque :  1
```

In [35]:
```
# Dada la cadena = "T7 Prom 5'-d CGC CAG GGT TTT TCA
# AGT CAC GAC GAC AGT TAT TGC TCA GCG G "
# 2- Pásalo a formato lista usando el espacio como separador
# palabras = cadena.split (" ")

cadena = "T7 Prom 5'-d CGC CAG GGT TTT TCA AGT CAC GAC GAC AGT TAT TGC TCA GCG G"

palabras = cadena.split(" ")

# 3- Muestra las distintas palabras del texto (pasa a conjuntos) (set)
# distintas =

distintas = set(palabras)

# 4- Ahora recorre distintas con un for, y responde (imprime) :
# 4.1- Cuántas veces aparece cada palabra
# 4.2- Cuántas palabras hay de 3 letras

cuenta = 0

for palabra in distintas :
     print (palabra, ":", cadena.count(palabra))
     if len(palabra) == 3:
        cuenta +=1

print ("hay", cuenta, "palabras de tres letras")


print (distintas)
```

```
TAT : 1
TGC : 1
G : 12
TTT : 1
CAC : 1
5'-d : 1
AGT : 2
GAC : 2
CAG : 1
GCG : 1
CGC : 1
TCA : 2
GGT : 1
Prom : 1
T7 : 1
hay 11 palabras de tres letras
{'TAT', 'TGC', 'G', 'TTT', 'CAC', "5'-d", 'AGT', 'GAC', 'CAG', 'GCG', 'CGC', 'TCA',
'GGT', 'Prom', 'T7'}
```

In [39]:
```
def saludar():
    print("Hola! Este print se llama desde la función saludar()")

saludar()
```

```
Hola! Este print se llama desde la función saludar()
```

In [1]:
```
# Ejercicio 7

# Escribir una función que reciba una muestra de números en una lista y devuelva ot


def quadrat (lista) :
    lista2 = []
    for i in lista:
        lista2.append(i**2)
    return lista2

print (quadrat ([1,2,3]))
```

```
[1, 4, 9]
```

In [47]:
```
def square(sample):
    """Función que calcula los cuadrados de una lista de números.
    Parámetros
    sample: Es una lista de números
    Devuelve una lista con los cuadrados de los números de la lista sample.
    """
    list = []
    for i in sample:
        list.append(i**2)
    return list

print(square([1, 2, 3, 4, 5]))
print(square([2.3, 5.7, 6.8, 9.7, 12.1, 15.6]))
```

```
[1, 4, 9, 16, 25]
[5.289999999999999, 32.49, 46.239999999999995, 94.08999999999999, 146.41, 243.35999
999999999]
```

In [2]:
```
# Escribir un programa que reciba una cadena de caracteres y devuelva un diccionari
# y su frecuencia.
# Escribir otra función que reciba el diccionario generado con la función anterior
# con la palabra más repetida y su frecuencia.

def contador_palabras(texto):

    texto = texto.split()
    palabras = {}

    for i in texto :
        if i in palabras:
            palabras[i] +=1
        else:
            palabras[i] = 1

    return palabras

texto = "Función que cuenta el número de veces que aparece cada palabra en un texto
print (contador_palabras(texto))
```

```
{'Función': 1, 'que': 2, 'cuenta': 1, 'el': 1, 'número': 1, 'de': 1, 'veces': 1, 'a
parece': 1, 'cada': 1, 'palabra': 1, 'en': 1, 'un': 1, 'texto': 1}
```

In [ ]:

```
def count_words(text):
    """Función que cuenta el número de veces que aparece cada palabra en un texto.
    Parámetros:
        - text: Es una cadena de caracteres.
    Devuelve:
        Un diccionario con pares palabra:frecuencia con las palabras contenidas en
    """
    text = text.split()
    words = {}
    for i in text:
        if i in words:
            words[i] += 1
        else:
            words[i] = 1
    return words

def most_repeated(words):
    max_word = ''
    max_freq = 0
    for word, freq in words.items():
        if freq > max_freq:
            max_word = word
            max_freq = freq
    return max_word, max_freq

text = 'Como quieres que te quiera si el que quiero que me quiera no me quiere como
print(count_words(text))
print(most_repeated(count_words(text)))
```