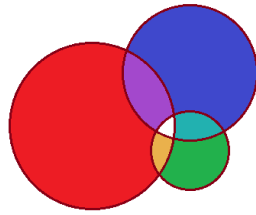


Множества



1. Описание/создание

In [3]:

```
A = {2, 4, 6, 8, 10, 12, 14, 16, 2}
B = {'Привет-привет!', 'Hello!', 'Привет', 3.1415}
print(A)
print(B)
print(type(A))
```

```
{2, 4, 6, 8, 10, 12, 14, 16}
{3.1415, 'Привет-привет!', 'Hello!', 'Привет'}
<class 'set'>
```

In []:

```
empty = set() # пустое множество
```

1.1 Из строк или списков

In [4]:

```
C = set('Привет-привет!')
D = set([1,2,3,4,5,1,2,3,4,5])
print(C, D)
```

```
{'т', 'п', 'и', 'в', '!', 'е', 'р', 'н', '-'} {1, 2, 3, 4, 5}
```

1.2 Генераторы

In [5]:

```
E = set(i for i in range(1,10) if i%2!=0)
F = {i for i in range(1,10) if i%2!=0}
print(E, F)
```

```
{1, 3, 5, 7, 9} {1, 3, 5, 7, 9}
```

Важно: элементы множества – неизменяемые. Не бывает множества списков.

In [6]:

```
A = {[1,2], [3,4]}
```

```
-----
TypeError                                Traceback (most recent call last)
<ipython-input-6-08ddf6cede0f> in <module>
----> 1 A = {[1,2], [3,4]}
```

```
TypeError: unhashable type: 'list'
```

2. Операции над множествами

Размер

In [7]:

```
# Размер (количество элементов)
n = len(A)
print(n)
```

8

In [8]:

```
# Добавление элемента
A.add(18)
print(A)
```

{2, 4, 6, 8, 10, 12, 14, 16, 18}

In [9]:

```
# Пример:
# альтернативное создание множества A
A = set()
for i in range(2,20,2):
    A.add(i)
print(A)
```

{2, 4, 6, 8, 10, 12, 14, 16, 18}

In [10]:

```
# Принадлежность элемента множеству
print(2 in A)
print('Привет' in B)
```

True

True

In [11]:

```
# создадим два множества
# для иллюстрации остальных операций
A = {i for i in range(11)}
B = {2**i for i in range(6)}
print(f'A = {A}\nB = {B}')
```

A = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10}

B = {32, 1, 2, 4, 8, 16}

In [12]:

```
# Объединение множеств
# способ 1
AB1 = A | B
# способ 2
AB2 = A.union(B)
print(f'A U B = {AB1}\nA U B = {AB2}')
```

A U B = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 32, 16}

A U B = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 32, 16}

In [13]:

```
# Пересечение множеств
# способ 1
AB1 = A & B
# способ 2
AB2 = A.intersection(B)
print(f'A ∩ B = {AB1}\nA ∩ B = {AB2}')
```

A ∩ B = {8, 1, 2, 4}

A ∩ B = {8, 1, 2, 4}

In [14]:

```
# Разность множеств
# способ 1
AB1 = A - B
# способ 2
AB2 = A.difference(B)
print(f'A \ B = {AB1}\nA \ B = {AB2}')
```

A \ B = {0, 3, 5, 6, 7, 9, 10}

A \ B = {0, 3, 5, 6, 7, 9, 10}

In [15]:

```
# Аналоги операций a += b, a -= b и т.п.

# удаляем из множества A все элементы,
# входящие в множество B
A -= B
# или так
A.difference_update(B)
print(A)

# добавляем в множество A
# всех элементов из множества B
A |= B
# или так
A.update(B)
print(A)

# оставляем в множестве A только
# те элементы, которые есть в B
A &= B
# или так
A.intersection_update(B)
print(A)

{0, 3, 5, 6, 7, 9, 10}
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 16, 32}
{32, 1, 2, 4, 8, 16}
```

In [16]:

```
A = {1,2,3,4,5}
B = {5,4,3,2,1}
# Проверка на вложение
print(A <= B)
# или так:
print(A.issubset(B))
print(A < B)
B = {5,4,3,2,1}
print(A < B)

# а можно и так:
A >= B
A.issuperset(B)
```

True
True
False
False

Out[16]:

True

Множество vs Список: скорость поиска

Как время поиска элемента в списке и множестве зависит от их размера?

In [17]:

```
# создадим большой список и большое множество
import time
n = 10**7
print('Создаем список...')
a = [i**2 for i in range(n)]
print('Создаем множество...')
b = set(a)
k1 = (n//2)**2 # этот элемент есть в списке/множестве
k2 = k1 + 1    # такого элемента нет
print('Готово!')
```

Создаем список...
Создаем множество...
Готово!

In [18]:

```
# Поиск в списке
t0 = time.time()
res = k1 in a
t1 = time.time()
print(f'Поиск в списке: ответ {res} получен за {t1-t0} сек')
t0 = time.time()
res = k2 in a
t1 = time.time()
print(f'Поиск в списке: ответ {res} получен за {t1-t0} сек')

# Поиск в множестве
t0 = time.time()
res = k1 in b
t1 = time.time()
print(f'Поиск в множестве: ответ {res} получен за {t1-t0} сек')
t0 = time.time()
res = k2 in b
t1 = time.time()
print(f'Поиск в множестве: ответ {res} получен за {t1-t0} сек')
```

Поиск в списке: ответ True получен за 0.2700004577636719 сек
Поиск в списке: ответ False получен за 0.49000072479248047 сек
Поиск в множестве: ответ True получен за 0.0 сек
Поиск в множестве: ответ False получен за 0.0 сек

Задачи по теме "Множества"

pythontutor.ru

In [19]:

```
# Для организации входных данных
from random import randint
```

Задача 1. Количество различных чисел

Дан список чисел. Определите, сколько в нем встречается различных чисел.

Примечание. Эту задачу на Питоне можно решить в одну строчку.

In [22]:

```
# Дано
a = [randint(-5,5) for i in range(12)]
print(a)
# Решение
print(len(set(a)))
```

[-4, 1, -2, 4, 1, -4, 1, -2, -3, 0, 0, 4]
6

Задача 2. Количество различных чисел

Даны два списка чисел. Посчитайте, сколько чисел содержится одновременно как в первом списке, так и во втором.

Примечание. Эту задачу на Питоне можно решить в одну строчку.

In [23]:

```
# Дано
a = [randint(-5,5) for i in range(12)]
print(a)
b = [randint(-5,5) for i in range(12)]
print(b)
# Решение
print(len(set(a) & set(b)))
```

[-2, -3, -4, 0, -2, -4, -1, 5, -2, -3, 4, 4]
[-5, 5, -2, 2, -3, 0, -1, -2, 5, 1, 5, 4]
6

Задача 3. Пересечение множеств

Даны два списка чисел. Найдите все числа, которые входят как в первый, так и во второй список и выведите их в порядке возрастания.

Примечание. И даже эту задачу на Питоне можно решить в одну строчку.

In [24]:

```
# Дано
a = [randint(-5,5) for i in range(12)]
print(a)
b = [randint(-5,5) for i in range(12)]
print(b)
# Решение
print(sorted(set(a)&set(b)))
```

```
[2, 2, 0, 2, -5, 0, 0, 1, -2, -2, 1, 1]
[-5, 4, -2, 2, -4, 1, -1, -1, 4, -2, -1, -1]
[-5, -2, 1, 2]
```

Задача 4. Количество слов в тексте

Дан текст: в первой строке записано число строк, далее идут сами строки. Определите, сколько различных слов содержится в этом тексте.

In [25]:

```
# решение
n = int(input('Кол-во строк:'))
words = set()
for i in range(n):
    words.update(set(input('> ').split()))
print(len(words), words)
```

```
Кол-во строк:3
> Задача 4. Количество слов в тексте
> Количество слов в тексте не велико
> сколько различных слов в тексте
10 {'слов', 'не', 'велико', 'в', 'сколько', '4.', 'Задача', 'тексте', 'различных', 'Количество'}
```

In []:

Leetcode.com

Сумма уникальных элементов. Дан список целых чисел `nums`. Найдите сумму тех элементов списка, которые встречаются в нем только один раз

Плохое правильное решение

In [26]:

```
def sum_of_unique(nums):
    s = 0
    for x in nums:
        if nums.count(x)==1:
            s += x
    return s
```

In [27]:

```
assert sum_of_unique([1,5,3,5])==4, 'Ошибка тестирования 1'
assert sum_of_unique([1,1,1,1])==0, 'Ошибка тестирования 2'
assert sum_of_unique([1,2,3,4])==10, 'Ошибка тестирования 3'
```

In [28]:

```
import random
N = 10**4
random.seed(123456)
nums = [random.randint(-N,N) for i in range(N)]
```

In [29]:

```
t = time.time()
assert sum_of_unique(nums)==191442, 'Ошибка тестирования 4'
dt = time.time()-t
print(dt)
assert dt<1, "Превышен допустимый интервал времени"
```

5.38000750541687

```
-----
AssertionError                                Traceback (most recent call last)
<ipython-input-29-4c720ae40ca8> in <module>
      3 dt = time.time()-t
      4 print(dt)
----> 5 assert dt<1, "Превышен допустимый интервал времени"
```

AssertionError: Превышен допустимый интервал времени

Правильное хорошее решение

In [33]:

```
def sum_of_unique(nums):
    a = set()
    b = set()# дубли
    for x in nums:
        if x in a:
            b.add(x)
        else:
            a.add(x)
    #print(a, b)
    return sum(a-b)
```

sum_of_unique([1,2,2,3, 5, 4, 1])

Out[33]:

12

In [34]:

```
assert sum_of_unique([1,2,3,2])==4, 'Ошибка тестирования 1'
assert sum_of_unique([1,1,1,1])==0, 'Ошибка тестирования 2'
assert sum_of_unique([1,2,3,4])==10, 'Ошибка тестирования 3'
```

In [35]:

```
t = time.time()
assert sum_of_unique(nums)==191442, 'Ошибка тестирования 4'
dt2 = time.time()-t
print(dt2)
assert dt2<1, "Превышен допустимый интервал времени"
```

0.009999990463256836

In [36]:

```
# Во сколько раз ускорили работу?
dt/dt2
```

Out[36]:

538.0012636196743

Один ряд клавиш. Дан список слов `words` . Выберите из него те слова, которые могут быть напечатаны, используя ровно один ряд русской клавиатуры (допускается также нажатие клавиш `Shift` или `CapsLock`)



In [37]:

```
def find_words(words):
    r1 = set('йцукенгшщзхъ')
    r2 = set('фывапролджэ')
    r3 = set('ячсмитьбю')
    res = []
    for w in words:
        s = set(w.lower())
        if s<=r1 or s<=r2 or s<=r3:
            res.append(w)
    return res
```

In [38]:

```
w1 = ['флора', 'форма', 'право', 'вектор',
      'эра', 'пора', 'ген', 'цинк',
      'чит', 'бить', 'порка', 'цена',
      'фара', 'плод', 'жить', 'ворон']
```

In [39]:

```
find_words(w1)
```

Out[39]:

```
['флора', 'право', 'эра', 'пора', 'ген', 'чит', 'бить', 'фара', 'плод']
```

In []: