

Практическая работа 6 - Множества

Множество (set) в Python - это неупорядоченная коллекция уникальных элементов. Важным свойством множества является то, что оно не допускает дубликатов элементов. Множества в Python поддерживают множество операций, таких как добавление элемента, удаление элемента, проверка наличия элемента, объединение множеств и другие.

Множества схожи со списками, но имеют ряд отличий.

Во-первых, множества создаются в абсолютно случайном порядке. Вы можете разместить элементы как вам будет угодно, но они все равно будут расположены впоследствии в случайном порядке.

Во-вторых, множества не могут иметь повторяющихся элементов. Все элементы с одинаковым значением не будут выведены повторно.

Множества удобно использовать когда вы хотите удалить повторяющиеся элементы из списка, например:

```
some_list = [12, 56, 91, 12]
set(some_list) # Результат: 12, 56, 91
```

Методы множества

- **set.update(other, ...)**; $set \mid= other \mid \dots$ - объединение.
- **set.intersection_update(other, ...)**; $set \&= other \& \dots$ - пересечение.
- **set.difference_update(other, ...)**; $set -= other \mid \dots$ - вычитание.
- **set.symmetric_difference_update(other)**; $set \wedge= other$ - множество из элементов, встречающихся в одном множестве, но не встречающиеся в обоих.
- **set.add(elem)** - добавляет элемент в множество.
- **set.remove(elem)** - удаляет элемент из множества. `KeyError`, если такого элемента не существует.
- **set.discard(elem)** - удаляет элемент, если он находится в множестве.
- **set.pop()** - удаляет первый элемент из множества. Так как множества не упорядочены, нельзя точно сказать, какой элемент будет первым.
- **set.clear()** - очистка множества.

- `len(s)` - число элементов в множестве (размер множества).
- `x in s` - принадлежит ли `x` множеству `s`.
- `set.isdisjoint(other)` - истина, если `set` и `other` не имеют общих элементов.
- `set == other` - все элементы `set` принадлежат `other`, все элементы `other` принадлежат `set`.
- `set.issubset(other)` или `set <= other` - все элементы `set` принадлежат `other`.
- `set.issuperset(other)` или `set >= other` - аналогично.
- `set.union(other, ...)` или `set | other | ...` - объединение нескольких множеств.
- `set.intersection(other, ...)` или `set & other & ...` - пересечение.
- `set.difference(other, ...)` или `set - other - ...` - множество из всех элементов `set`, не принадлежащие ни одному из `other`.
- `set.symmetric_difference(other)`; `set ^ other` - множество из элементов, встречающихся в одном множестве, но не встречающиеся в обоих.
- `set.copy()` - копия множества.

Frozenset

Frozenset - метод, что позволяет создать, которое нельзя изменять в ходе выполнения программы. Получается, что Frozenset это смесь множества и кортежа.

Вот основные операции и характеристики множеств в Python:

1) Создание множества:

Множество можно создать, перечислив его элементы в фигурных скобках `{}` или с помощью конструктора `set()`. Например:

```
my_set = {1, 2, 3}
```

2) Добавление элементов:

Элементы можно добавлять в множество с помощью метода `add()`. Например:

```
my_set.add(4)
```

3) Удаление элементов:

Элементы можно удалять из множества с помощью метода `remove()` или `discard()`. Разница между ними в том, что `remove()` вызовет ошибку, если элемент не найден, а `discard()` просто ничего не сделает. Например:

```
my_set.remove(3)
```

4) Проверка наличия элемента:

Можно проверить, содержит ли множество определенный элемент с помощью оператора `in`. Например:

```
if 2 in my_set:  
    print("2 есть во множестве")
```

5) Длина множества:

Длину множества можно получить с помощью функции len(). Например:

```
length = len(my_set)
```

6) Операции с множествами:

Множества поддерживают операции объединения, пересечения, разности и др. Например:

```
set1 = {1, 2, 3}  
set2 = {3, 4, 5}
```

```
union_set = set1 | set2 # Объединение  
intersection_set = set1 & set2 # Пересечение  
difference_set = set1 - set2 # Разность
```

7) Итерация по множеству:

Можно итерироваться по элементам множества с использованием цикла for. Например:

```
for item in my_set:  
    print(item)
```

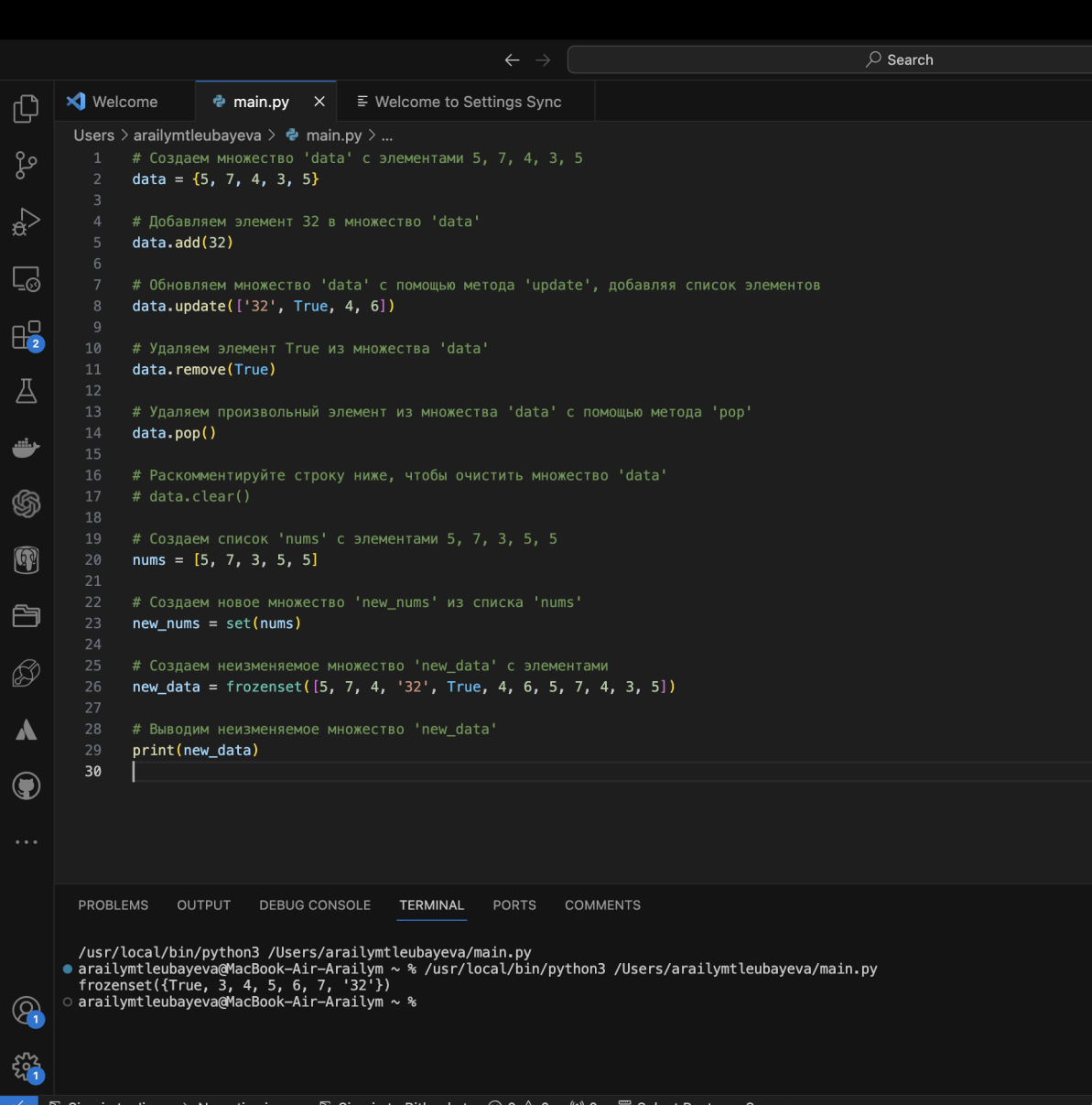
8) Преобразование списка в множество и наоборот:

Вы можете преобразовать список в множество с помощью set() и множество в список с помощью list(). Например:

```
my_list = [1, 2, 3, 1, 2]  
my_set = set(my_list) # Преобразование списка в множество, удаляя дубликаты
```

Множества полезны, когда вам нужно хранить уникальные значения и выполнять операции над ними эффективно. Они являются важной частью стандартной библиотеки Python и могут использоваться в различных сценариях, включая обработку данных и удаление дубликатов.

Пример - Множества



```
1 # Создаем множество 'data' с элементами 5, 7, 4, 3, 5
2 data = {5, 7, 4, 3, 5}
3
4 # Добавляем элемент 32 в множество 'data'
5 data.add(32)
6
7 # Обновляем множество 'data' с помощью метода 'update', добавляя список элементов
8 data.update(['32', True, 4, 6])
9
10 # Удаляем элемент True из множества 'data'
11 data.remove(True)
12
13 # Удаляем произвольный элемент из множества 'data' с помощью метода 'pop'
14 data.pop()
15
16 # Раскомментируйте строку ниже, чтобы очистить множество 'data'
17 # data.clear()
18
19 # Создаем список 'nums' с элементами 5, 7, 3, 5, 5
20 nums = [5, 7, 3, 5, 5]
21
22 # Создаем новое множество 'new_nums' из списка 'nums'
23 new_nums = set(nums)
24
25 # Создаем неизменяемое множество 'new_data' с элементами
26 new_data = frozenset([5, 7, 4, '32', True, 4, 6, 5, 7, 4, 3, 5])
27
28 # Выводим неизменяемое множество 'new_data'
29 print(new_data)
30 |
```

PROBLEMS OUTPUT DEBUG CONSOLE **TERMINAL** PORTS COMMENTS

```
/usr/local/bin/python3 /Users/arailymtleubayeva/main.py
● arailymtleubayeva@MacBook-Air-Arilym ~ % /usr/local/bin/python3 /Users/arailymtleubayeva/main.py
frozenset({True, 3, 4, 5, 6, 7, '32'})
○ arailymtleubayeva@MacBook-Air-Arilym ~ %
```

Самостоятельная работа

1) Работа с множествами

Создайте множество состоящее из 5 элементов.

Выполните такие операции как:

- удалите элемент со значением 67
- добавьте два новых элемента
- выведите множество на экран

2) Очистка повторений

Есть список:

```
list = [1, 53, 8, 9, 34, 1, 0, 53, 53, 8, 73, 5]
```

Удалите из него все повторяющиеся элементы, преобразовав его в множество.

3) Создание множества

Создайте множество при помощи:

- функции set;
- фигурных скобок;
- при помощи цикла for.

4) Объединение

Создайте set и frozenset. Объедините оба множества в одно целое.

Выполните операции:

- к объединенному множеству добавьте элемент 2 и 5;
- удалите число 2, а также первый элемент в множестве.