Python: множества | Я.Шпора

Множество (set) в Python — изменяемая неупорядоченная коллекция.
Элементы множества — уникальные значения неизменяемых типов.

Особенности множества

- В множество можно добавлять элементы и удалять их.
- К элементам множества нельзя обратиться по индексам или автоматически вывести их по порядку.
- Нельзя изменить отдельный элемент множества.
- Элементы множества должны быть уникальны.

> Множества — это не последовательности: к элементам множества нельзя обратиться по индексам, порядок элементов в множестве не определён и при каждом обращении может быть разным.

Как объявить множество

С помощью фигурных скобок

```
tomato_ratings = {5.0, 4.1, 4.3, 4.7, 4.7, 3.8}
print(tomato_ratings)
# Вывод в терминал: {5.0, 4.3, 3.8, 4.1, 4.7}

print(type(tomato_ratings))
# Вывод в терминал: <class 'set'>
```

> Объявить пустое множество через запись `empty = {}` не получится: будет создан пустой словарь.

С помощью функции `set()`

```
# Вызов функции set() без аргументов создаст пустое множество.
```

```
empty = set()
print('Содержимое переменной empty:', empty)
print('Тип переменной empty:', type(empty))
# Вывод в терминал:
# Содержимое переменной empty: set()
# Тип переменной empty: <class 'set'>

# Создать список.
vegetables = ['Помидоры', 'Огурцы', 'Баклажаны', 'Перец', 'Капуста']

# Преобразовать список во множество.
vegetables_set = set(vegetables)
```

> Элементами множества не могут быть данные изменяемых типов, например списки.

Работа со множествами

Добавить элемент

Для добавления элемента в множество применяют `set.add()`:

```
vegetables = {'томат', 'огурец', 'баклажан', 'лук'}

set.add(vegetables, 'капуста')
print('Изменённое множество:', vegetables)
# Вывод в терминал:
# Изменённое множество: {'огурец', 'томат', 'лук', 'баклажан', 'капуста'}
```

Удалить элемент

Удалить элемент из множества можно тремя способами: `set.remove(<set>, element)`, `set.discard(<set>, element)` и `set.pop(<set>, element)`.

Операции `set.remove()` и `set.discard()` очень схожи. Разница между ними в том, как они ведут себя при попытке удалить элемент, которого нет во множестве.

Если попытаться удалить несуществующий элемент:

- `set.remove()` выдаст ошибку, выполнение программы остановится;
- `set.discard()` проигнорирует эту попытку и не вызовет ошибку.

```
vegetables = {'томат', 'огурец', 'баклажан', 'лук'}

set.remove(vegetables, 'баклажан')
print('Удалили элемент "Баклажан":', vegetables)

set.discard(vegetables, 'лук')
print('Удалили элемент "Лук"', vegetables)
# Вывод в терминал: {'томат', 'огурец'}

set.discard(vegetables, 'картошка')
print('Попытались удалить несуществующий элемент через set.discard():', ve
# Вывод в терминал: {'томат', 'огурец'}

set.remove(vegetables, 'картошка')
print(vegetables)
# Вывод в терминал: KeyError: 'картошка'
```

`set.pop()` удаляет и возвращает случайный элемент множества:

```
vegetables = {'томат', 'огурец', 'баклажан', 'лук'}
vegetable = vegetables.pop()
print(f'Сегодня на обед - {vegetable}!')
print(f'Оставшиеся овощи: {vegetables}')
# Вывод в терминал:
# Сегодня на обед - томат!
# Оставшиеся овощи: {'огурец', 'лук', 'баклажан'}
```

Очистить множество

Для очистки существующего множества используется `set.clear()` :

```
vegetables = {'томат', 'огурец', 'баклажан', 'лук'}
set.clear(vegetables)
print(vegetables)
# Вывод в терминал: set()
```

Проверить наличие элемента в множестве

Чтобы определить, есть ли нужный элемент в множестве, применяют оператор `in` :

```
vegetables = set(['Помидор', 'Огурец', 'Баклажан'])
if 'Помидор' in vegetable_yields:
    print('Помидор есть!')
else:
    print('Помидора нет!')

# Вывод в терминал: Помидоры есть!
```

Операции над множествами

Пересечение: оператор `&` (логическое И)

```
stepan_vegetables = {'томат', 'огурец', 'баклажан', 'лук'}
tonya_vegetables = {'огурец', 'кабачок', 'баклажан', 'морковь'}

common_vegetables = stepan_vegetables & tonya_vegetables
print(common_vegetables)
# Вывод в терминал: {'огурец', 'баклажан'}
```

Объединение: оператор `|` (логическое ИЛИ)

```
stepan_vegetables = {'томат', 'огурец', 'баклажан', 'лук'}
tonya_vegetables = {'огурец', 'кабачок', 'баклажан', 'морковь'}

all_vegetables = stepan_vegetables | tonya_vegetables
print(all_vegetables)
# Вывод в терминал: {'томат', 'огурец', 'баклажан', 'лук', 'кабачок', 'мо
```

Разность: оператор `-`

```
stepan_vegetables = {'томат', 'огурец', 'баклажан', 'лук'}
tonya_vegetables = {'огурец', 'кабачок', 'баклажан', 'морковь'}

stepan_unique_vegetables = stepan_vegetables - tonya_vegetables
print(stepan_unique_vegetables)
# Вывод в терминал: {'томат', 'лук'}
```

Симметричная разность: оператор `^`

```
stepan_vegetables = {'томат', 'огурец', 'баклажан', 'лук'}
tonya_vegetables = {'огурец', 'кабачок', 'баклажан', 'морковь'}

sym_diff_vegetables = stepan_vegetables ^ tonya_vegetables
print(sym_diff_vegetables)
# Вывод в терминал: {'томат', 'лук', 'кабачок', 'морковь'}
```

На пути к истине мудрый признаёт
необходимость шпаргалок, ведущих
его сквозь пустыню забытого.

Чинь Ин, Хранитель Императорских Шпор