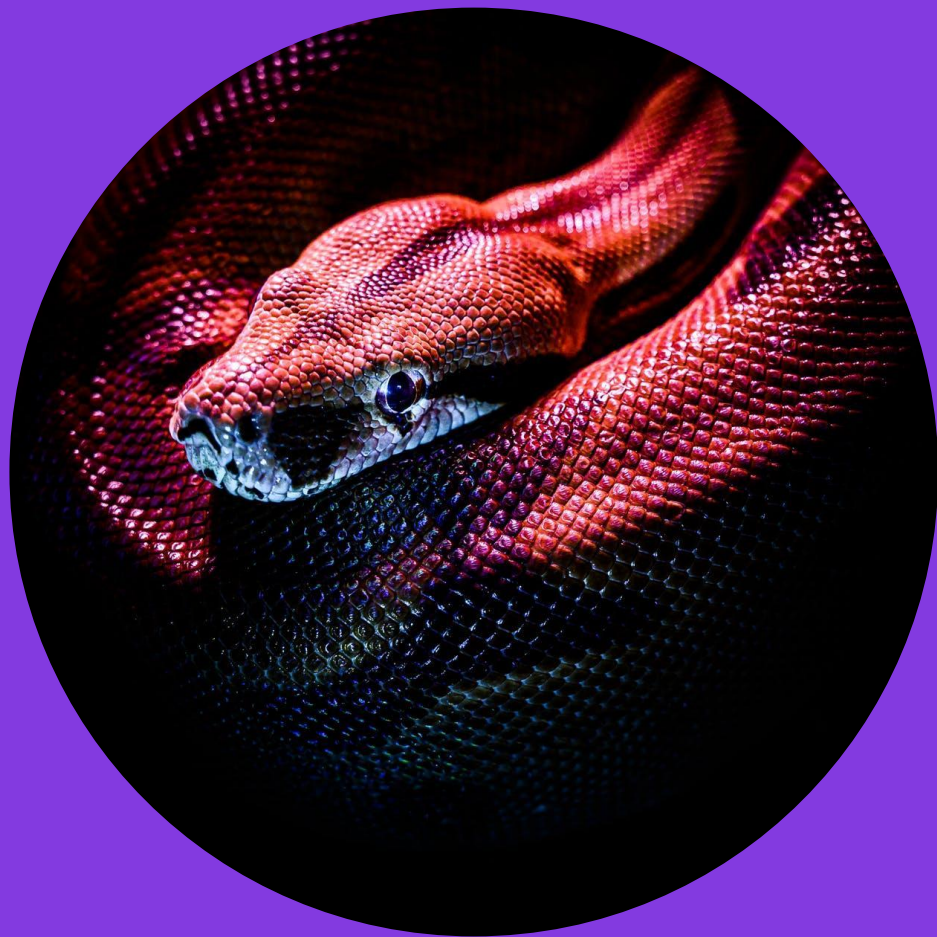


алгоритмика

Модуль 7. Урок 2.

# Множества



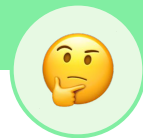
[Методические указания](#)

# Во время урока откройте презентацию в режиме Просмотр (ctrl+enter).

Так ученики увидят только свои слайды. В презентации есть методические слайды, которые нужны только преподавателю. Они отмечены иконкой «глаз».

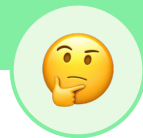


**Слайд-инструкция**



### Задача:

**написать программу, которая хранит список различных книг, прочитанных одним классом.**



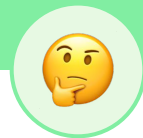
**С помощью каких инструментов Python  
можно запомнить прочитанные книги?**



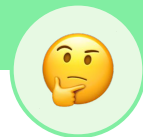
## Хранение прочитанных книг

**Список книг**

**Словарь с книгами**

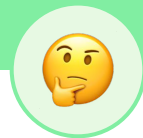


**По какому алгоритму будет работать программа, хранящая прочитанные книги в списке?**



### Возможная схема программы со списками:

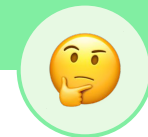
- Пользователь вводит название книги.
- Программа просматривает список, есть ли книга с таким названием.
- Если такой книги нет, то она добавляется в список.



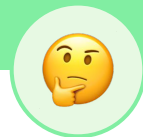
### Возможная схема программы со словарями:

- Пользователь вводит название книги.
- Программа сразу определяет, есть ли книга в словаре среди ключей (оператор `in`).
- Если такой книги нет, то создаётся новая пара «ключ-элемент».



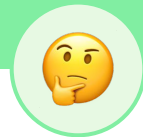


**В чём преимущества и недостатки  
каждого способа?**



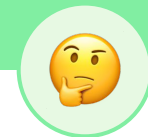
**Программа со списками:**

**книги могут дублироваться  
(неэкономный расход памяти).**

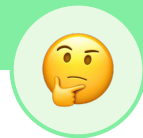


**Программа со словарями:**

**нужно хранить лишние данные,  
кроме названия книги  
(не эффективно).**



**Какая структура данных  
решила бы задачу наиболее  
эффективно?**



**Для базы данных книг нам нужна:**

**структура данных,**

**хранящая элементы только по 1 разу,**

**порядок расположения элементов и их  
номера не важны.**

Такой инструмент есть в Python!



**Множество** —  
**это неупорядоченный набор элементов.**

**Каждый элемент** встречается в множестве  
**только 1 раз.**



# Создание множества

```
many = set()
```



Команда, создающая  
пустое множество



Название множества



## Создание множества

many = {1, 2, 3}



Элементы множества



Название множества





## Пример:

### Программа

```
many = {1, 2, 3, 3, 2, 1}  
print(many)
```

### Вывод

```
{1, 2, 3}
```

**Элементы множества  
не повторяются**



# Множества удобно использовать, если:



**важно знать входит элемент во множество или нет;**



**другие свойства элемента (например, порядковый номер) не важны.**



# Добавление элемента во множество

`many.add(5)`



Добавляемый элемент



Метод, добавляющий элемент,  
указанный в скобках



## Пример:

### Программа

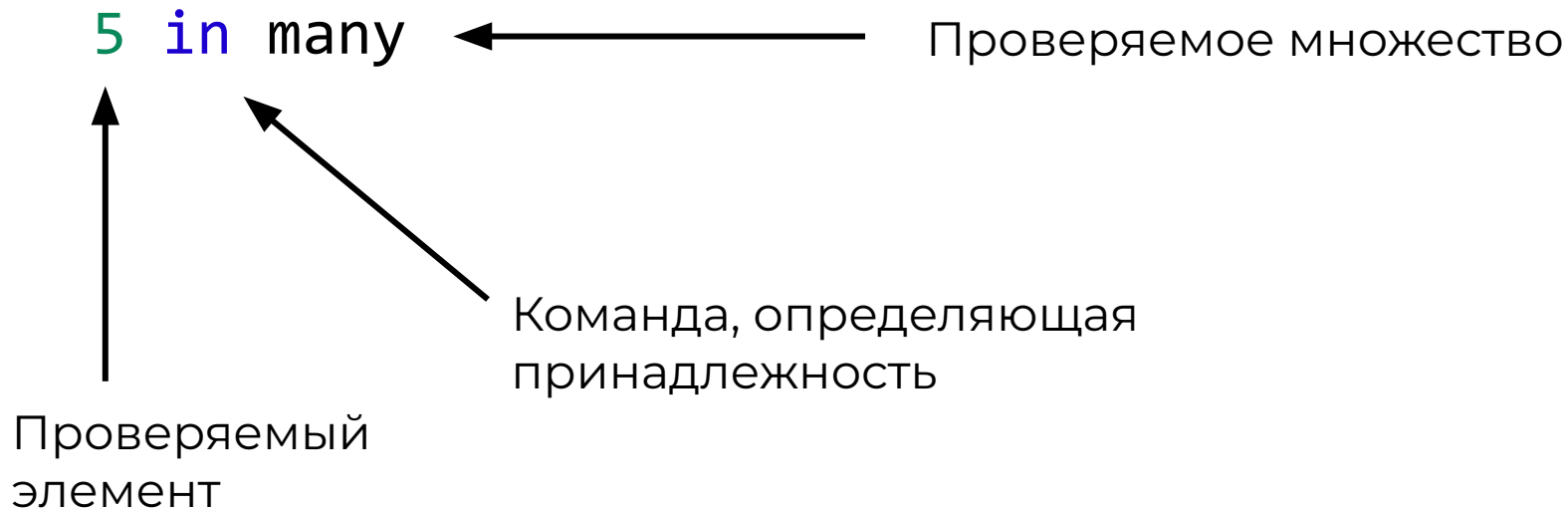
```
many = {1, 2, 3}  
many.add(5)  
print(many)
```

### Вывод

```
{1, 2, 3, 5}
```



## Принадлежит ли элемент множеству





## Пример:

### Программа

```
many = {1, 2, 3, 5}
if 5 in many:
    print("Yes!")
```

### Вывод

Yes



**Примеры и  
программа,**  
хранящая названия  
прочитанных книг.



# Работаем на платформе



**1 - заходим  
на платформу.**

**2 - выбираем  
задание  
«Множества:  
задачи».**







**Перерыв**



**Делимся на команды  
по 3 или 4 человека.**

**Каждая команда  
выбирает капитана.**



## Ход игры:

 **Команда читает вопрос на экране.**

 **Команда придумывает ответ (2 минуты).**

 **Капитан записывает ответ на листе.**



**Что такое  
список?**



**Назови 2 любых отличия  
списка от множества.**



# Какая это структура данных?

```
phones={ 'Ваня' : 9067345, 'Коля' : 916224 }
```



**Что делает метод `add()`?**



**Что делает оператор `in`?**





# Что напечатает программа?

```
numbers = [1, 2, 3, 0]  
print(numbers[0])  
numbers.remove(1)  
print(numbers)
```



**Сдай лист с ответом учителю.**

**Получи лист другой команды.**

**Отметь правильные ответы и  
посчитай баллы.**



**А теперь ответы!**



**Список** — это упорядоченный набор элементов.



- **Элементы в списке упорядочены, во множестве нет.**
- **Каждый элемент списка имеет номер (индекс), а во множестве нет.**
- **Элементы в списке могут повторяться.**



## Это словарь

```
phones={ 'Ваня' : 9067345, 'Коля' : 916224 }
```

**Ключ**



**Значение**





**Метод** `add()` **добавляет элемент**  
**в множество.**



**Оператор `in` определяет,  
входит ли элемент в словарь  
или множество.**





### Программа:

```
numbers = [1, 2, 3, 0]
print(numbers[0])
numbers.remove(1)
print(numbers)
```

### Вывод:

```
1
[2, 3, 0]
```



**Подведём итоги**

# Работаем на платформе



**1 - заходим  
на платформу.**

**2 - выбираем  
задание  
«Структуры данных:  
задачи».**





**До скорых встреч!**