1. ***Лабораторная работа.* Множества.**

Множество в языке Питон — это структура данных, эквивалентная множествам в математике. Множество может состоять из различных элементов, порядок элементов в множестве неопределен. В множество можно добавлять и удалять элементы, можно перебирать элементы множества, можно выполнять операции над множествами (объединение, пересечение, разность). Можно проверять принадлежность элемента множеству.

В отличие от массивов, где элементы хранятся в виде последовательного списка, в множествах порядок хранения элементов неопределен (более того, элементы множества хранятся не подряд, а при помощи различных алгоритмов). Это позволяет выполнять операции типа “проверить принадлежность элемента множеству” быстрее, чем просто перебирая все элементы множества.

Элементами множества может быть любой неизменяемый тип данных: числа, строки, кортежи. Изменяемые типы данных не могут быть элементами множества, в частности, нельзя сделать элементом множества список (но можно сделать кортеж) или другое множество. Требование неизменяемости элементов множества накладывается особенностями представления множества в памяти компьютера.

## Задание множеств

Множество задается перечислением всех его элементов в фигурных скобках. Исключением является пустое множество, которое можно создать при помощи функции set(). Если функции set передать в качестве параметра список, строку или кортеж, то она вернёт множество, составленное из элементов списка, строки, кортежа. Например:

A = {1, 2, 3}

A = set('qwerty')

print(A)

выведет {'e', 'q', 'r', 't', 'w', 'y'}.

Каждый элемент может входить в множество только один раз, порядок задания элементов неважен. Например, программа:

A = {1, 2, 3}

B = {3, 2, 3, 1}

print(A == B)

выведет True, так как A и B — равные множества.

Каждый элемент может входить в множество только один раз. set('Hello') вернет множество из четырех элементов: {'H', 'e', 'l', 'o'}.

## Работа с элементами множеств

Узнать число элементов в множестве можно при помощи функции len.

Перебрать все элементы множества (в неопределенном порядке!) можно при помощи цикла for:

primes = {2, 3, 5, 7, 11}

for num in primes:

print(num)

Проверить, принадлежит ли элемент множеству можно при помощи операции in, возвращающей значение типа bool. Аналогично есть противоположная операция not in. Для добавления элемента в множество есть метод add:

A = {1, 2, 3}

print(1 in A, 4 not in A)

A.add(4)

Для удаления элемента x из множества есть два метода: discard и remove. Их поведение различается только в случае, когда удаляемый элемент отсутствует в множестве. В этом случае метод discard не делает ничего, а метод remove генерирует исключение KeyError.

Наконец, метод pop удаляет из множества один случайный элемент и возвращает его значение. Если же множество пусто, то генерируется исключение KeyError.

Из множества можно сделать список при помощи функции list.

**Операции с множествами**

С множествами в питоне можно выполнять обычные для математики операции над множествами.

|  |  |
| --- | --- |
| **A | B A.union(B)** | Возвращает множество, являющееся объединением множеств A и B. |
| **A |= B A.update(B)** | Добавляет в множество A все элементы из множества B. |
| **A & B A.intersection(B)** | Возвращает множество, являющееся пересечением множеств A и B. |
| **A &= B A.intersection\_update(B)** | Оставляет в множестве A только те элементы, которые есть в множестве B. |
| **A - B A.difference(B)** | Возвращает разность множеств A и B (элементы, входящие в A, но не входящие в B). |
| **A -= B A.difference\_update(B)** | Удаляет из множества A все элементы, входящие в B. |
| **A ^ B A.symmetric\_difference(B)** | Возвращает симметрическую разность множеств A и B (элементы, входящие в A или в B, но не в оба из них одновременно). |
| **A ^= B A.symmetric\_difference\_update(B)** | Записывает в A симметрическую разность множеств A и B. |
| **A <= B A.issubset(B)** | Возвращает true, если A является подмножеством B. |
| **A >= B A.issuperset(B)** | Возвращает true, если B является подмножеством A. |
| **A < B** | Эквивалентно A <= B and A != B |
| **A > B** | Эквивалентно A >= B and A != B |

* 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.

И операции, непосредственно изменяющие множество:

* **set.update**(other, ...); set |= other | ... - объединение.
* **set.intersection\_update**(other, ...); set &= other & ... - пересечение.
* **set.difference\_update**(other, ...); set -= other | ... - вычитание.
* **set.symmetric\_difference\_update**(other); set ^= 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.

**frozenset**

Единственное отличие set от frozenset заключается в том, что set - изменяемый тип данных, а frozenset - нет.

**ЗАДАНИЕ**

1. Организовать процедуру для задания элементов множества А. Организовать процедуру printset, печатающую элементы множества А с определением их числа.
2. Описать множество В. Ввести значения его элементов. Распечатать множество B с помощью процедуры printset.
3. Задать два текстовых предложения. Сравнить при помощи множеств эти предложения, не используя стандартные операция для работы со множествами (такие как + - \* /) и найти общие слова, встречающиеся в обоих предложениях.
4. Задать множество S состоящее из символов. Разделить множество S на три подмножества: А – символы русского алфавита, Б – латинские символы, В – цифры.