

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ  
ФЕДЕРАЦИИ  
ФГАОУ ВО «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»  
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ**

**Кафедра инфокоммуникаций**

**Отчет  
по лабораторной работе №5  
«Работа с множествами в языке Python»  
по дисциплине:  
«Введение в системы искусственного интеллекта»**

**Вариант 4**

Выполнил: студент группы ИВТ-б-о-18-1 (2)  
Дрищёв Данила Николаевич

\_\_\_\_\_ (подпись)

Проверил:  
Воронкин Роман Александрович

\_\_\_\_\_ (подпись)

Ставрополь, 2022 г.

**Цель работы:** приобретение навыков по работе с множествами при написании программ с помощью языка программирования Python версии 3.x.

## Задание 1

```
Ввод [3]: 1 if __name__ == "__main__":
2           u = set("abcdefghijklmnopqrstuvwxyz")
3
4           a = {"a", "b", "h", "j", "l"}
5           b = {"b", "c", "h", "l", "r", "v"}
6           c = {"j", "k", "n", "t", "z"}
7           d = {"b", "i", "k", "v", "w"}
8
9
10
11          x = (a.union(b)).intersection(c)
12          print(f"x = {x}")
13
14          an = u.difference(a)
15          bn = u.difference(b)
16
17          y = (an.intersection(bn)).difference(c.union(d))
18          print(f"y = {y}")
19
20
x = {'j'}
y = {'u', 'd', 'o', 'y', 'g', 'x', 'q', 'e', 's', 'm', 'p', 'f'}
```

$$A = \{a, b, h, j, l\}; \quad B = \{b, c, h, l, r, v\}; \quad C = \{j, k, n, t, z\}; \quad D = \{b, i, k, v, w\};$$
$$X = (A \cup B) \cap C; \quad Y = (\bar{A} \cap \bar{B}) / (C \cup D).$$

Ввод [ ]: 1

Рисунок 1 – Листинг программы

**Вывод:** в ходе выполнения работы были приобретены навыки по работе с множествами при написании программ с помощью языка программирования Python версии 3.x.

Файлы с решением данных задач находится на **Github** :

<https://github.com/drishchevd/drishchev>

## 1. Что такое множества в языке Python?

Множества в Python – это структура данных, которые содержат неупорядоченные элементы. Элементы также не являются индексированным. Как и список, множество позволяет внесение и удаление элементов.

Однако, есть ряд особенных характеристик, которые определяют и отделяют множество от других структур данных:

Множество не содержит дубликаты элементов;

Элементы множества являются неизменными (их нельзя менять), однако само по себе множество является изменяемым, и его можно менять;

Так как элементы не индексируются, множества не поддерживают никаких операций среза и индексирования.

## **2. Как осуществляется создание множеств в Python?**

Существует два пути, следуя которым, мы можем создавать множества в Python.

Мы можем создать множество путем передачи всех элементов множества внутри фигурных скобок {} и разделить элементы при помощи запятых (.). Множество может содержать любое количество элементов и элементы могут быть разных типов, к примеру, целые числа, строки, кортежи, и т. д. Однако, множество не поддерживает изменяемые элементы, такие как списки, словари, и так далее.

Мы также можем создать множество из списков. Это можно сделать, вызвав встроенную функцию Python под названием set().

## **3. Как проверить присутствие/отсутствие элемента в множестве?**

Операция `x in sets` проверяет наличие значения элемента `x` в множестве `sets`. Если значение `x` присутствует в множестве операция вернет `True`, если нет, то `False`.

Операция `x not in sets` проверяет отсутствие значения элемента `x` в множестве `sets`. Если значение `x` присутствует в множестве операция вернет `False`, если нет, то `True`.

Эта операция поддерживается как изменяемыми `set`, так и неизменяемыми множествами `frozenset`

## **4. Как выполнить перебор элементов множества?**

При помощи цикла `for` можно перебрать все элементы множества:

```
Primes = {2, 3, 5, 7, 11}
```

```
for num in Primes:
```

```
    print(num)
```

## **5. Что такое set comprehension?**

Представления списков в Python используются очень часто. Но это не единственный тип представлений. Вы также можете создавать представления множеств и словарей (set comprehension и dictionary comprehension). set comprehension почти ничем не отличается от представления списка. Разница лишь в том, что заданные значения обеспечивают, чтобы выходные данные не содержали дубликатов. Вы можете создать set comprehension, используя фигурные скобки вместо квадратных:

## **6. Как выполнить добавление элемента во множество?**

Python позволяет нам вносить новые элементы во множество при помощи функции add(). Например:

```
Python
months = set(["Jan", "March", "Apr", "May", "June", "July", "Aug", "Sep",
"Oct", "Nov", "Dec"])
months.add("Feb")
print(months)
```

## **7. Как выполнить удаление одного или всех элементов множества?**

Python позволяет нам удалять элемент из множества, но не используя индекс, так как множество элементов не индексированы. Элементы могут быть удалены при помощи обоих методов discard() и remove().

Помните, что метод discard() не будет выдавать ошибку, если элемент не был найден во множестве. Однако, если метод remove() используется и элемент не был найден, возникнет ошибка.

## **8. Как выполняются основные операции над множествами: объединение, пересечение, разность?**

Объединение множеств

Противоположная операция – объединение двух множеств выполняется с помощью оператора |:

```
setA = {1,2,3,4}
setB = {3,4,5,6,7}
setA | setB
```

### Пересечение множеств

Для любых двух множеств:

```
setA = {1,2,3,4}
```

```
setB = {3,4,5,6,7}
```

можно вычислять их пересечение, то есть, находить значения, входящие в состав обоих множеств. Это делается с помощью оператора &:

```
setA & setB
```

### Вычитания множеств

Следующая операция – это вычитание множеств. Например, для множеств:

```
setA = {1,2,3,4}
```

```
setB = {3,4,5,6,7}
```

```
setA - setB
```

### Сравнение множеств

Множества можно сравнивать между собой:

На равенство

```
setA == setB
```

На неравенство

Противоположное сравнение на неравенство записывается так:

```
setA != setB
```

На больше, меньше

В Python операторы <, > применительно к множествам, по сути, определяют вхождение или не вхождение одного множества в другое.

## **9. Как определить, что некоторое множество является надмножеством или подмножеством другого множества?**

Функция `issubset()` предназначена для определения того, есть ли текущее множество подмножеством другого. Общая форма объявления функции следующая

```
set.issubset(other)
```

где

set – текущее множество;

other – другое множество которое сравнивается на вхождение с множеством set.

Функция `issubset()` возвращает значение `True`, если выполняется одно из условий:

множество set есть подмножеством множества other. Иными словами, все элементы множества set есть в множестве other;

множества set и other равны между собой.

В других случаях функция `issubset()` возвращает значение `False`. Функция `issubset()` может быть заменена операцией `<=`.

функция `issuperset()` определяет, есть ли текущее множество надмножеством над другим. Общая форма объявления функции следующая

`set.superset(other)`

где

set – текущее множество;

other – другое множество, которое сопоставляется с множеством set.

Если set есть надмножеством над other, то функция возвращает `True`. В противном случае функция возвращает `False`.

Функция `issuperset()` может быть заменена операцией `>=`.

## **10. Каково назначение множеств `frozenset` ?**

`Frozenset` (замороженное множество) – это класс с характеристиками множества, однако, как только элементы становятся назначенными, их нельзя менять. Кортежи могут рассматриваться как неизменяемые списки, в то время как `frozenset`-ы — как неизменные множества.

Множества являются изменяемыми и хешируемыми, это значит, что мы не можем использовать их как словарные ключи. Замороженные множества (`frozenset`) являются хешированными и могут использоваться в качестве ключей словаря.

Для создания замороженного множества, мы используем метод `frozenset()`.

## **11. Как осуществляется преобразование множеств в строку, список, словарь?**

Для преобразования в строку используется функция `str ()`. Аргументом функции `str ()` может выступать число, строка, кортеж, список, множество, словарь, логическое значение, `None`. Любой объект, преобразованный в строку, становится просто набором символов.