

Федеральное государственное автономное образовательное учреждение
высшего образования «Северо-Кавказский федеральный университет»
Институт цифрового развития

ОТЧЕТ ПО ДИСЦИПЛИНЕ
ОСНОВЫ ПРОГРАММНОЙ ИНЖЕНЕРИИ
Лабораторная работа №10
Работа с множествами в языке Python

Выполнила:

Мирзаева Камилла Мирзаевна

2 курс, группа ПИЖ-б-о-20-1

Принял:

Воронкин Роман Александрович

Ставрополь, 2021 г.

Ход работы:

1. Способ создания множества

```
>>> a = {1, 2, 0, 1, 3, 2}
>>> print(a)
{0, 1, 2, 3}
```

2. Способ создания множества с помощью команды set

```
>>> a = set('data')
>>> print(a)
{'d', 'a', 't'}
```

3. Проверка присутствия/ отсутствия значений с помощью команды in/not in

```
>>> a = {0, 1, 2, 3}
>>> print(2 in a)
True
>>> a = {0, 1, 2, 3}
>>> print(2 not in a)
False
```

4. Перебор всех элементов путем команды for

```
for a in {0, 1, 2}:  
    print(a)
```

for a in {0, 1, 2}

modul1 ×

C:\Users\79616\anaconda3\python.exe

0
1
2

5. Генерация множеств

```
a = {i for i in [1, 2, 0, 1, 3, 2]}  
print(a)
```

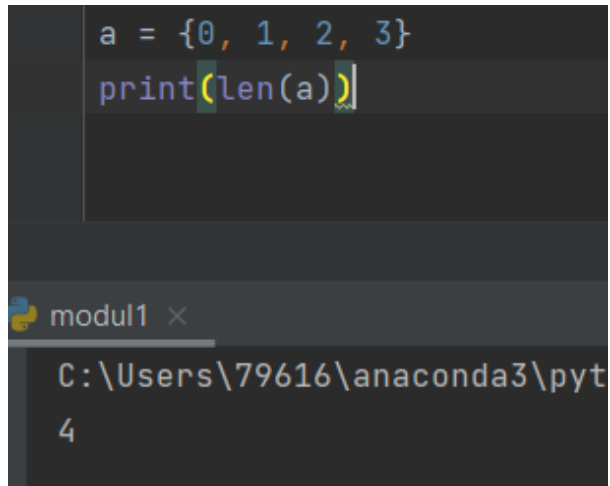
modul1 ×

C:\Users\79616\anaconda3\python.exe D:/УЧЕБ.

{0, 1, 2, 3}

6. Получение размера – метод len

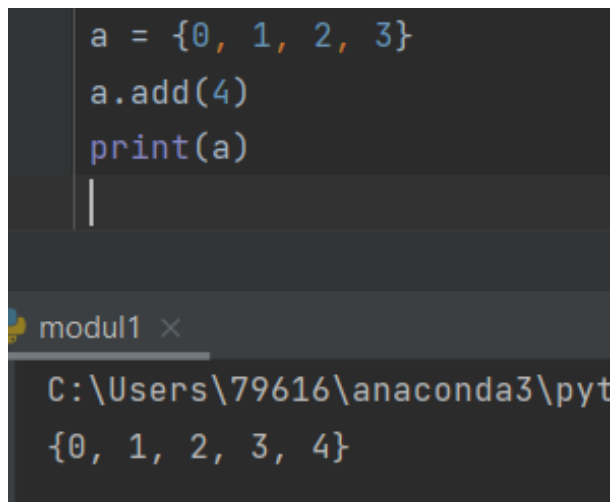
```
a = {0, 1, 2, 3}
print(len(a))
```



The screenshot shows a code editor with two lines of Python code: `a = {0, 1, 2, 3}` and `print(len(a))`. Below the code, the output of the program is displayed in a terminal window, showing the number 4.

7. Добавление элемента путем команды add

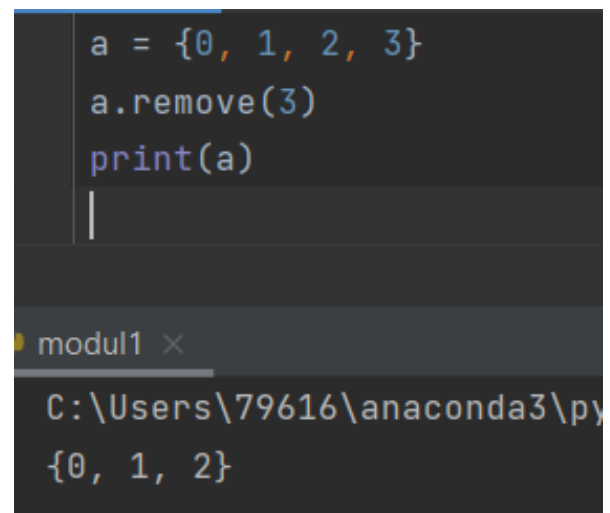
```
a = {0, 1, 2, 3}
a.add(4)
print(a)
```



The screenshot shows a code editor with three lines of Python code: `a = {0, 1, 2, 3}`, `a.add(4)`, and `print(a)`. Below the code, the output of the program is displayed in a terminal window, showing the set `{0, 1, 2, 3, 4}`.

8. Удаление элемента с помощью команды remove

```
a = {0, 1, 2, 3}
a.remove(3)
print(a)
```



The screenshot shows a code editor with three lines of Python code: `a = {0, 1, 2, 3}`, `a.remove(3)`, and `print(a)`. Below the code, the output of the program is displayed in a terminal window, showing the set `{0, 1, 2}`.

9. Полная очистка – метод clear

```
a = {0, 1, 2, 3}
a.clear()
print(a)
```

modul1 ×

C:\Users\79616\anaconda3\python.exe
set()

10. Порядок элементов в множестве

```
a = {0, 1, 12, 'b', 'ab', 3, 2, 'a'}
print(a)
```

modul1 ×

C:\Users\79616\anaconda3\python.exe D:/УЧЕБА/0
{0, 1, 2, 3, 'a', 12, 'ab', 'b'}

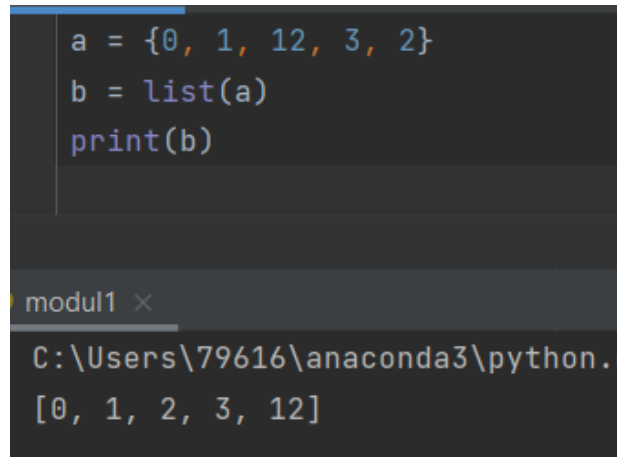
```
a = {0, 1, 12, 3, 2}
print(a)
```

modul1 ×

C:\Users\79616\anaconda3\python.exe
{0, 1, 2, 3, 12}

11.Преобразование упорядоченного множества в список

```
a = {0, 1, 12, 3, 2}
b = list(a)
print(b)
```

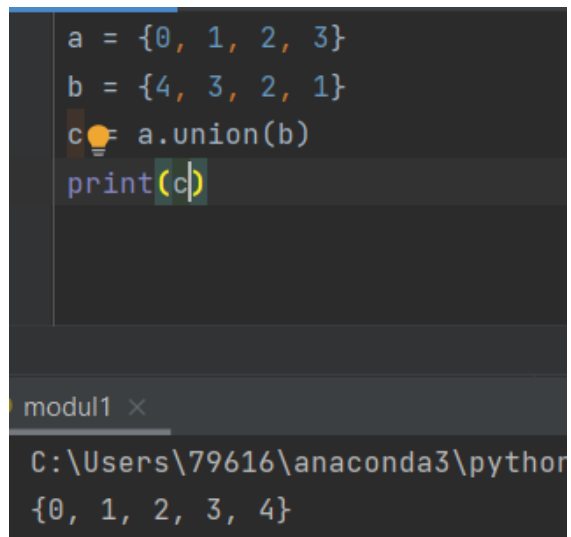


modul1 ×

C:\Users\79616\anaconda3\python.
[0, 1, 2, 3, 12]

12.Операция объединения - union

```
a = {0, 1, 2, 3}
b = {4, 3, 2, 1}
c = a.union(b)
print(c)
```

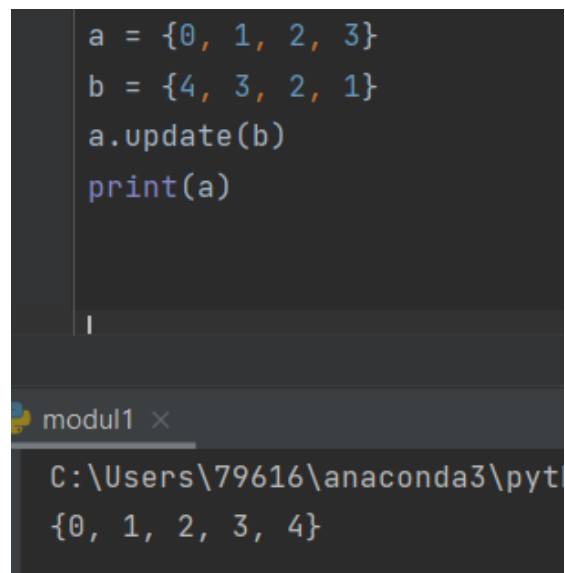


modul1 ×

C:\Users\79616\anaconda3\python
{0, 1, 2, 3, 4}

13.Операция добавления - update

```
a = {0, 1, 2, 3}
b = {4, 3, 2, 1}
a.update(b)
print(a)
```



modul1 ×

C:\Users\79616\anaconda3\pyt
{0, 1, 2, 3, 4}

14. Операция пересечения - intersection

```
a = {0, 1, 2, 3}
b = {4, 3, 2, 1}
c = a.intersection(b)
print(c)
```

modul1 ×

C:\Users\79616\anaconda3\python.exe
{1, 2, 3}

15. Операция разности - difference

```
a = {0, 1, 2, 3}
b = {4, 3, 2, 1}
c = a.difference(b)
print(c)
```

modul1 ×

C:\Users\79616\anaconda3\python.exe
{0}

16. Определения подмножества и надмножества

```
>>> a = {0, 1, 2, 3, 4}
>>> b = {3, 2, 1}
>>> print(a.issubset(b))
False
>>> a = {0, 1, 2, 3, 4}
>>> b = {3, 2, 1}
>>> print(a.issuperset(b))
True
```

17. Тип frozenset

```
a = frozenset({"Hello", "world!"})
print(a)
```

modul1 ×

C:\Users\79616\anaconda3\python.exe D:/YЧЕ
frozenset({'Hello', 'world!'})

18. Преобразование множеств в строку

```
a = {'set', 'str', 'dict', 'list'}
b = ','.join(a)
print(b)
print(type(b))
```

modul1 ×

C:\Users\79616\anaconda3\python.exe D:/Y
list,str,dict,set
<class 'str'>

19.Преобразование множеств в словарь

```
a = {('a', 2), ('b', 4)}
b = dict(a)
print(b)
print(type(b))
```

modul1 ×

C:\Users\79616\anaconda3\python.exe D:
{'b': 4, 'a': 2}
<class 'dict'>

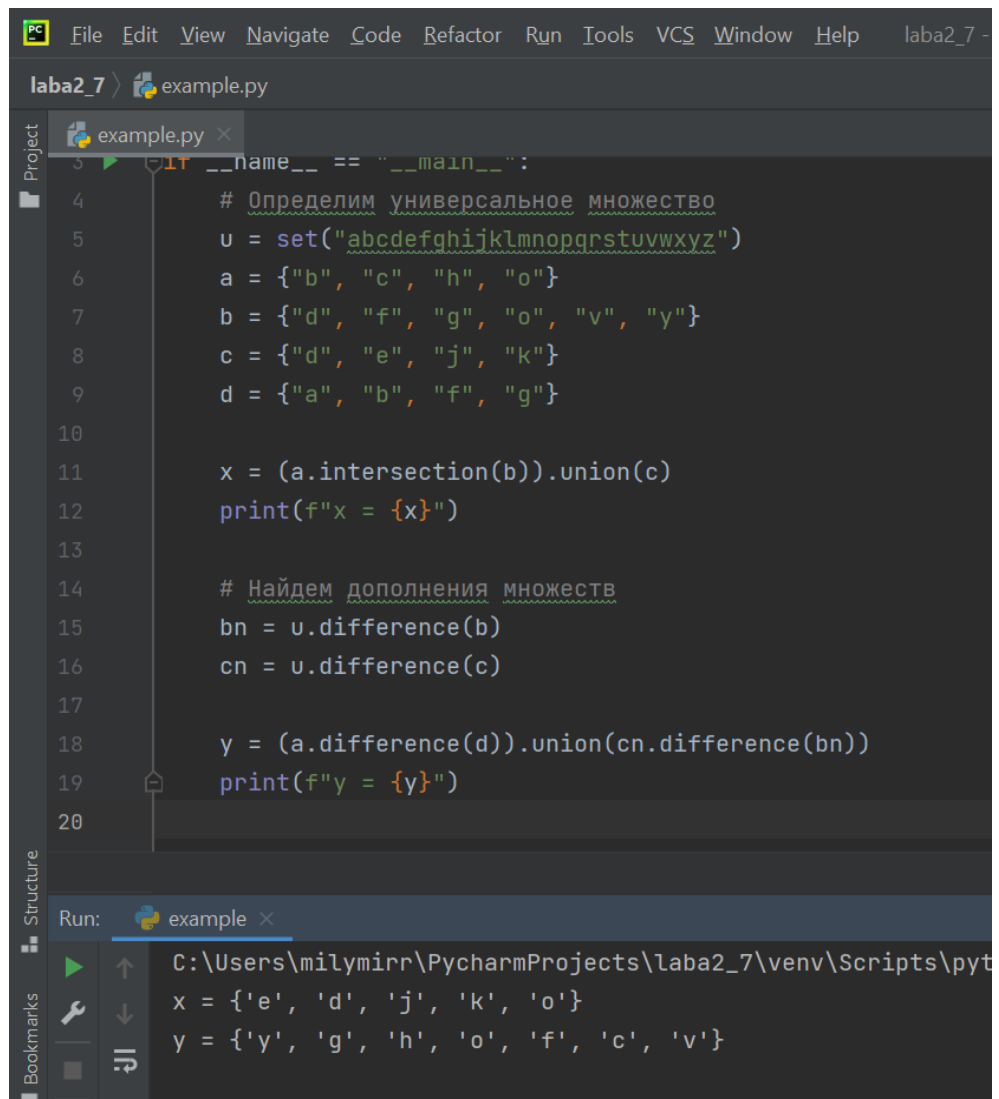
20.Преобразование множеств в список

```
a = {1, 2, 0, 1, 3, 2}
b = list(a)
print(b)
print(type(b))
```

modul1 ×

C:\Users\79616\anaconda3\python.exe
[0, 1, 2, 3]
<class 'list'>

21.Пример № 1



The screenshot shows the PyCharm IDE interface. The main editor window displays a Python script named `example.py` with the following code:

```
1 if __name__ == "__main__":
2     # Определим универсальное множество
3     u = set("abcdefghijklmnopqrstuvwxyz")
4     a = {"b", "c", "h", "o"}
5     b = {"d", "f", "g", "o", "v", "y"}
6     c = {"d", "e", "j", "k"}
7     d = {"a", "b", "f", "g"}
8
9
10
11     x = (a.intersection(b)).union(c)
12     print(f"x = {x}")
13
14     # Найдем дополнения множеств
15     bn = u.difference(b)
16     cn = u.difference(c)
17
18     y = (a.difference(d)).union(cn.difference(bn))
19     print(f"y = {y}")
20
```

The left sidebar shows the Project and Structure tool windows. The bottom panel shows the Run console with the output of the script:

```
Run: example x
C:\Users\milymirr\PycharmProjects\laba2_7\venv\Scripts\python.exe
x = {'e', 'd', 'j', 'k', 'o'}
y = {'y', 'g', 'h', 'o', 'f', 'c', 'v'}
```

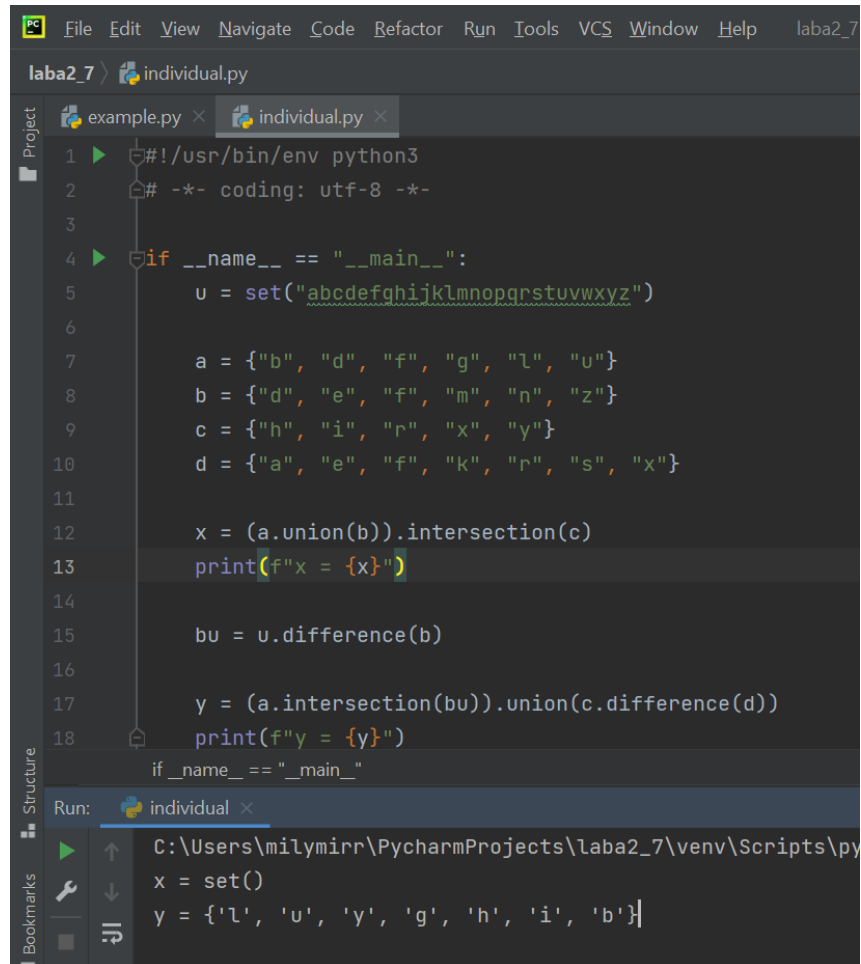
Индивидуальное задание

Вариант 15

15.

$$X = (A \cup B) \cap C; \quad Y = (A \cap \bar{B}) \cup (C/D).$$

$$A = \{b, d, f, g, l, u\}; \quad B = \{d, e, f, m, n, z\}; \quad C = \{h, i, r, x, y\}; \quad D = \{a, e, f, k, r, s, x\};$$



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 if __name__ == "__main__":
5     u = set("abcdefghijklmnopqrstuvwxyz")
6
7     a = {"b", "d", "f", "g", "l", "u"}
8     b = {"d", "e", "f", "m", "n", "z"}
9     c = {"h", "i", "r", "x", "y"}
10    d = {"a", "e", "f", "k", "r", "s", "x"}
11
12    x = (a.union(b)).intersection(c)
13    print(f"x = {x}")
14
15    bu = u.difference(b)
16
17    y = (a.intersection(bu)).union(c.difference(d))
18    print(f"y = {y}")
19
20 if __name__ == "__main__":
```

Run: individual ×

```
C:\Users\milymirr\PycharmProjects\laba2_7\venv\Scripts\py
x = set()
y = {'l', 'u', 'y', 'g', 'h', 'i', 'b'}
```

Контрольные вопросы:

1. Что такое множества в языке Python? Множеством в Python называется неупорядоченная совокупность уникальных значений. В качестве элементов этого набор данных могут выступать любые неизменяемые объекты, такие как числа, символы, строки.
2. Как осуществляется создание множеств в Python? Сделать это можно, просто присвоив переменной последовательность значений, выделив их фигурными скобками.

```
a = {1, 2, 0, 1, 3, 2}
print(a)

{0, 1, 2, 3}
```

Существует и другой способ создания множеств, который подразумевает использование вызова set.

```
a = set('data')
print(a)

{'d', 'a', 't'}
```

3. Как проверить присутствие/отсутствие элемента в множестве?

- Проверка, есть ли данное значение в множестве. Для этого используется in.

```
a = {0, 1, 2, 3}
print(2 in a)

True
```

- Наоборот, проверка отсутствия. Используется not in.

```
a = {0, 1, 2, 3}
print(2 not in a)

False
```

4. Как выполнить перебор элементов множества?

- Перебор всех элементов.

```
for a in {0, 1, 2}:
    print(a)

0
1
2
```

5. Что такое set comprehensions?

Для создания множества можно в Python воспользоваться генератором, позволяющих заполнять списки, а также другие наборы данных с учетом неких условий.

```
a = {i for i in [1, 2, 0, 1, 3, 2]}  
print(a)  
  
{0, 1, 2, 3}
```

6. Как выполнить добавление элемента во множество?

Чтобы внести новые значения, потребуется вызывать метод `add`.

7. Как выполнить удаление одного или всех элементов множества?

Для удаления элементов из множества используются следующие функции в Python

`remove` — удаление элемента с генерацией исключения в случае, если такого элемента нет;

`discard` — удаление элемента без генерации исключения, если элемент отсутствует;

`pop` — удаление первого элемента, генерируется исключение при попытке удаления из пустого множества.

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

Чтобы объединить все элементы двух разных множеств, стоит воспользоваться методом `union` на одном из объектов.

Чтобы найти общие элементы для двух разных множеств, следует применить функцию `intersection`.

Чтобы вычислить разность для двух разных множеств, необходимо воспользоваться методом `difference`.

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

Чтобы выяснить, является ли множество `a` подмножеством `b`, стоит попробовать вывести на экран результат выполнения метода `issubset`,

Чтобы узнать, является ли множество `a` надмножеством `b`, необходимо вызвать метод `issuperset` и вывести результат его работы на экран.

10. Каково назначение множеств `frozenset`?

Множество, содержимое которого не поддается изменению имеет тип `frozenset`. Значения из этого набора нельзя удалить, как и добавить новые.

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

Строка. Для преобразования множества в строку используется конкатенация текстовых значений, которую обеспечивает функция `join`. В этом случае ее аргументом является набор данных в виде нескольких строк. Запятая в кавычках выступает в качестве символа, разделяющего значения. Метод `type` возвращает тип данных объекта в конце приведенного кода.

```
a = {'set', 'str', 'dict', 'list'}
b = ','.join(a)
print(b)
print(type(b))

set,dict,list,str
<class 'str'>
```

Словарь. Чтобы получить из множества словарь, следует передать функции `dict` набор из нескольких пар значений, в каждом из которых будет находиться ключ. Функция `print` демонстрирует на экране содержимое полученного объекта, а `type` отображает его тип.

```
a = {('a', 2), ('b', 4)}
b = dict(a)
print(b)
print(type(b))

{'b': 4, 'a': 2}
<class 'dict'>
```

Список. На этот раз используется вызов `list`, получающий в качестве аргумента множество `a`. На выходе функции `print` отображаются уникальные значения для изначального набора чисел.

```
a = {1, 2, 0, 1, 3, 2}
b = list(a)
print(b)
print(type(b))

[0, 1, 2, 3]
<class 'list'>
```