

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ**

**Отчет по лабораторной работе №10
«Работа с множествами в языке Python»**

по дисциплине «Основы программной инженерии»

Выполнила:
Первых Дарья Александровна,
2 курс, группа ПИЖ-б-о-20-1,
Проверил:
Доцент кафедры инфокоммуникаций,
Воронкин Р.А.

Ставрополь, 2021 г.

ВЫПОЛНЕНИЕ

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

main ×

C:\Users\podar\study\anacon

{0, 1, 2, 3}

Рисунок 1 – Пример создания множества целых чисел

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

main ×

C:\Users\podar\study\an

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

Рисунок 2 – Пример создания множества с помощью метода set

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

main ×

C:\Users\podar\study

True

Рисунок 3 – Пример проверки, есть ли данное значение в множестве

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

main ×

C:\Users\podar\study\an

False

Рисунок 4 – Пример проверки отсутствия значения в множестве

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

for a in {0, 1, 2}

main ×

C:\Users\podar\study\ana

0

1

2

Рисунок 5 – Пример перебора элементов

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

main ×

C:\Users\podar\study\anaconda\envs\10

{0, 1, 2, 3}

Рисунок 6 – Пример генерации множества

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

main ×

C:\Users\podar\study\an

4

Рисунок 7 – Пример получения размера

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

main ×

C:\Users\podar\study\anacon

{0, 1, 2, 3, 4}

Рисунок 8 – Пример добавления элемента

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

main ×

C:\Users\podar\study\ana

{0, 1, 2}

Рисунок 9 – Пример удаления элемента

```
1 a = {0, 1, 2, 3}
2 a.clear()
3 print(a)
4
5 set()
```

main ×

C:\Users\podar\stud

set()

Рисунок 10 – Пример полной очистки

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

main ×

C:\Users\podar\study\anaconda\envs\10LRV

{0, 1, 2, 3, 'b', 12, 'ab', 'a'}

Рисунок 11 – Пример порядка элементов в множестве

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

main ×

C:\Users\podar\study\ana

{0, 1, 2, 3, 12}

Рисунок 12 – Пример порядка чисел в множестве

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

main ×

C:\Users\podar\study\anacon

[0, 1, 2, 3, 12]

Рисунок 13 – Пример порядка элементов в множестве, преобразованном в список

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

main ×

C:\Users\podar\study\anacon

{0, 1, 2, 3, 4}

Рисунок 14 – Пример операции объединения

```
1 a = {0, 1, 2, 3}
2 b = {4, 3, 2, 1}
3 a.update(b)
4 print(a)
5
```

main ×

C:\Users\podar\study\anacon

{0, 1, 2, 3, 4}

Рисунок 15 – Пример операции добавления

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

main ×

C:\Users\podar\study\anac

{1, 2, 3}

Рисунок 16 – Пример операции пересечения

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

main ×

C:\Users\podar\study\anac

{0}

Рисунок 17 – Пример операции разность

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

main ×

C:\Users\podar\study\anac

False

Рисунок 18 – Пример определения подмножества

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

main x

C:\Users\podar\study\anaconda

True

Рисунок 19 – Пример определения надмножества

```
1 a = frozenset({"hello", "world"})
2 print(a)
```

main x

C:\Users\podar\study\anaconda\envs\10

frozenset({'hello', 'world'})

Рисунок 20 – Пример использования типа frozenset

```
1 a = {'set', 'str', 'dict', 'list'}
2 b = ','.join(a)
3 print(b)
4 print(type(b))
5
```

main x

C:\Users\podar\study\anaconda\envs\10

list,dict,str,set

<class 'str'>

Рисунок 21 – Пример преобразования множества в строку

```
1 a = {('a', 2), ('b', 4)}
2 b = dict(a)
3 print(b)
4 print(type(b))
5
```

main ×

C:\Users\podar\study\anacon

{'b': 4, 'a': 2}

<class 'dict'>

Рисунок 22 – Пример преобразования множества в словарь

```
1 a = {1, 2, 0, 1, 3, 2}
2 b = list(a)
3 print(b)
4 print(type(b))
5
```

main ×

C:\Users\podar\study\anacon

[0, 1, 2, 3]

<class 'list'>

Рисунок 23 – Пример преобразования множества в список


```
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 ▶ if __name__ == "__main__":
4     # Определим универсальное множество
5     u = set("abcdefghijklmnopqrstuvwxyz")
6     a = {"b", "c", "h", "o"}
7     b = {"d", "f", "g", "o", "v", "y"}
8     c = {"d", "e", "j", "k"}
9     d = {"a", "b", "f", "g"}
10
11     x = (a.intersection(b)).union(c)
12     print(f"x = {x}")
13
14     # Найдем дополнения множеств
15     bn = u.difference(b)
```

main ×

C:\Users\podar\study\anaconda\envs\10LR\python.
x = {'d', 'j', 'k', 'e', 'o'}
y = {'h', 'g', 'f', 'v', 'c', 'y', 'o'}

Рисунок 24 – Пример решения задачи: определить результат выполнения операции над множествами. Считать элементы множества строками.

```
4 import sys
5
6 if __name__ == "__main__":
7     u = set("aeiouy")
8     x = input("Введите слово: ")
9     count = 0
10    for letter in x:
11        if letter in u:
12            count += 1
13    print(f"Количество гласных равно: {count}")
```

if __name__ == "__main__"

main x

C:\Users\podar\study\anaconda\envs\10LR\python.exe

Введите слово: *you do such a great job*

Количество гласных равно: 9

```
1 # !/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import sys
5
6 if __name__ == "__main__":
7     u = set("aeiouy")
8     x = input("Введите слово: ")
9     count = 0
10    for letter in x:
11        if letter in u:
12            count += 1
13    print(f"Количество гласных равно: {count}")
```

if __name__ == "__main__"

main x

C:\Users\podar\study\anaconda\envs\10LR\python.exe

Введите слово: *you can do it*

Количество гласных равно: 6

Рисунок 25 – Пример решения задания №1

```
1  #!/usr/bin/env python3
2  #- coding: utf-8 #-
3
4  import sys
5
6  if __name__ == "__main__":
7      x = set(input("Введите первую строку: "))
8      y = set(input("Введите вторую строку: "))
9      common_letters = x.intersection(y)
10     print(', '.join(common_letters))
```

if __name__ == "__main__"

main x

C:\Users\podar\study\anaconda\envs\10LR\python.exe "C:\Users\podar\study\anaconda\envs\10LR\python.exe"

Введите первую строку: you beautiful

Введите вторую строку: u wonderful

e, , o, f, l, u

```
1  #!/usr/bin/env python3
2  #- coding: utf-8 #-
3
4  import sys
5
6  if __name__ == "__main__":
7      x = set(input("Введите первую строку: "))
8      y = set(input("Введите вторую строку: "))
9      common_letters = x.intersection(y)
10     print(', '.join(common_letters))
```

if __name__ == "__main__"

main x

C:\Users\podar\study\anaconda\envs\10LR\python.exe "C:\Users\podar\study\anaconda\envs\10LR\python.exe"

Введите первую строку: you do it

Введите вторую строку: you can do this

, o, u, i, d, y, t

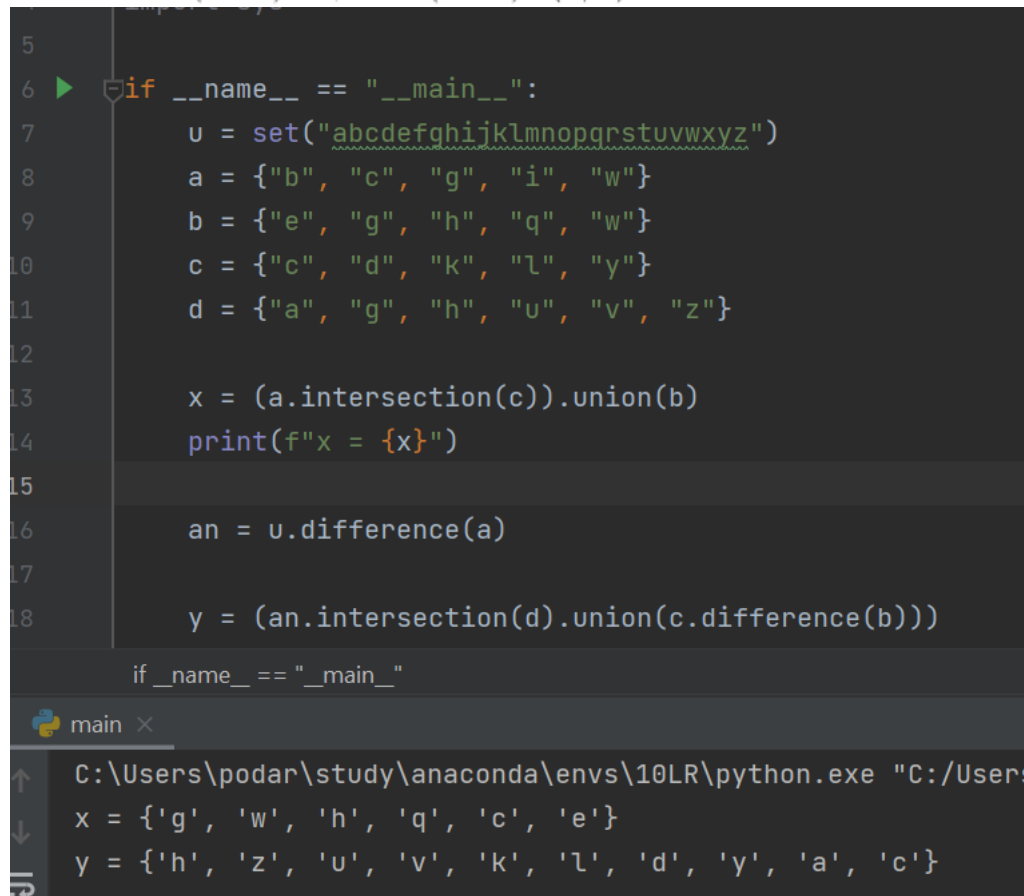
Рисунок 26 – Пример решения задания №2

Индивидуальное задание.

$A = \{b, c, g, l, w\}; \quad B = \{e, g, h, q, w\}; \quad C = \{c, d, k, l, y\}; \quad D = \{a, g, h, u, v, z\};$

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

```
5
6 ▶ if __name__ == "__main__":
7     u = set("abcdefghijklmnopqrstuvwxyz")
8     a = {"b", "c", "g", "l", "w"}
9     b = {"e", "g", "h", "q", "w"}
10    c = {"c", "d", "k", "l", "y"}
11    d = {"a", "g", "h", "u", "v", "z"}
12
13    x = (a.intersection(c)).union(b)
14    print(f"x = {x}")
15
16    an = u.difference(a)
17
18    y = (an.intersection(d).union(c.difference(b)))
19
20    if __name__ == "__main__"
```



```
C:\Users\podar\study\anaconda\envs\10LR\python.exe "C:/Users\podar\study\anaconda\envs\10LR\python.exe"
x = {'g', 'w', 'h', 'q', 'c', 'e'}
y = {'h', 'z', 'u', 'v', 'k', 'l', 'd', 'y', 'a', 'c'}
```

Рисунок 27 – Пример решения индивидуального задания.

Вопросы

1. Что такое множества в языке Python?

Множество называется неупорядоченная совокупность уникальных значений. В качестве элементов набора данных могут выступать неизменяемые объекты, такие как числа, символы, строки.

2. Как осуществляется создание множеств в Python?

Присвоить переменной последовательность значений, выделив их фигурными скобками.

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

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

Вызвать set.

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

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

3. Как проверить присутствие/отсутствие элемента в множестве?

Проверка присутствия:

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

True
```

Проверка отсутствия:

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

False
```

4. Как выполнить перебор элементов множества?

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

0
1
2
```

5. Что такое set comprehension?

Set comprehensions – генератор, позволяющий заполнять списки, а также другие наборы данных с учётом 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`. Аргументом в данном случае будет добавляемый элемент последовательности.

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

{0, 1, 2, 3, 4}
```

7. Как выполнить удаление одного или всех элементов множества?

`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.

`Clear` – полная очистка.

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

Объединение:

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

{0, 1, 2, 3, 4}
```

Пересечение:

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

{1, 2, 3}
```

Разность:

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

{0}
```

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

Определение подмножества:

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

False
```

Определения надмножества:

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

True
```

10. Каково назначение множеств frozenset?

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

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

frozenset({'hello', 'world'})
```

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

Для преобразования множества в строку используется конкатенация текстовых значений, которую обеспечивает функция join. В этом случае её 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 приведённого кода.

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

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

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'>
```

По аналогии с предыдущими преобразованиями можно получить список неких объектов. 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'>
```