

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное
образовательное учреждение высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Отчет по лабораторной работе № 2.7

Работа с множествами в языке Python

Выполнил студент группы ИВТ-б-о-20-1

Погорелов Д.Н

Работа защищена « » _____ 20__ г.

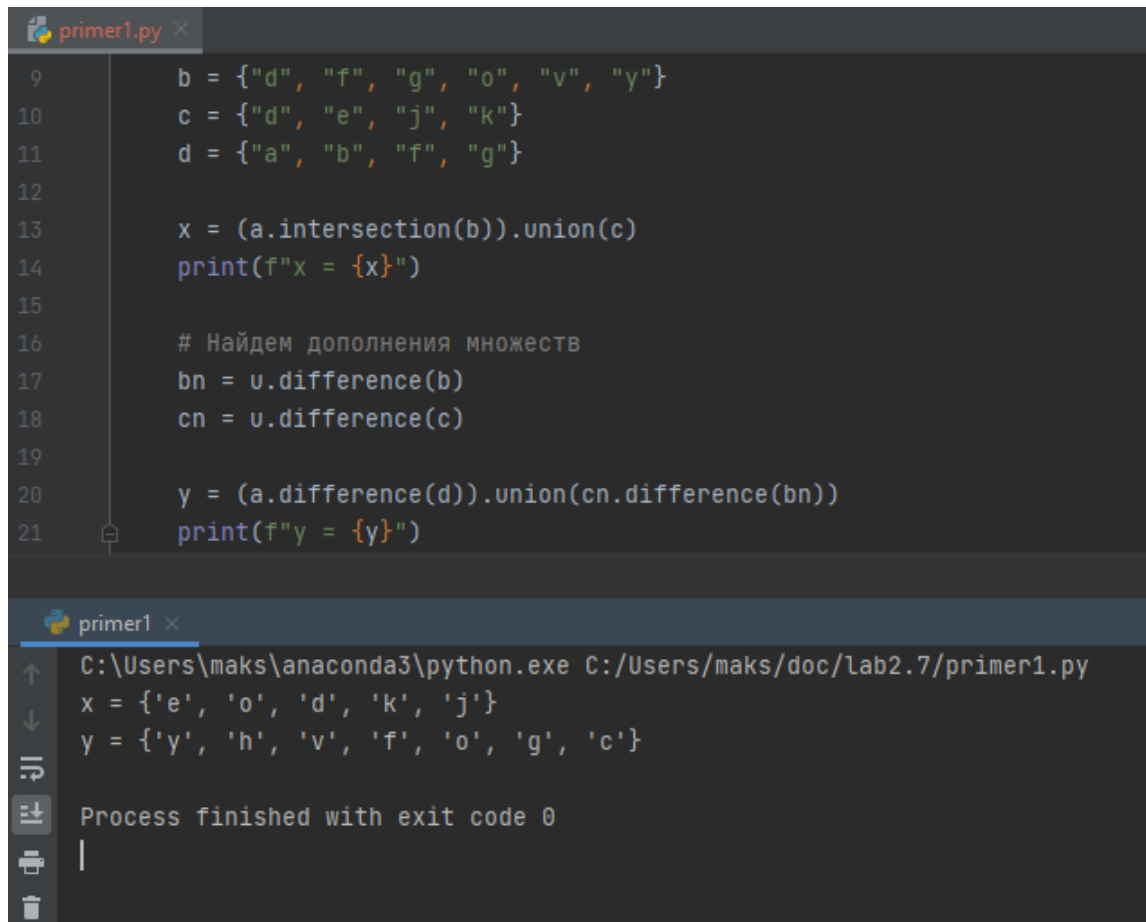
Проверил(а) _____

Ставрополь 2021

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

1. Создан общедоступный репозиторий на GitHub. Дополнен файл .gitignore необходимыми правилами для работы с IDE PyCharm.

2. Проработан пример из лабораторной работы, рис. 1



The screenshot displays the PyCharm IDE interface. The top pane shows a Python script named `primer1.py` with the following code:

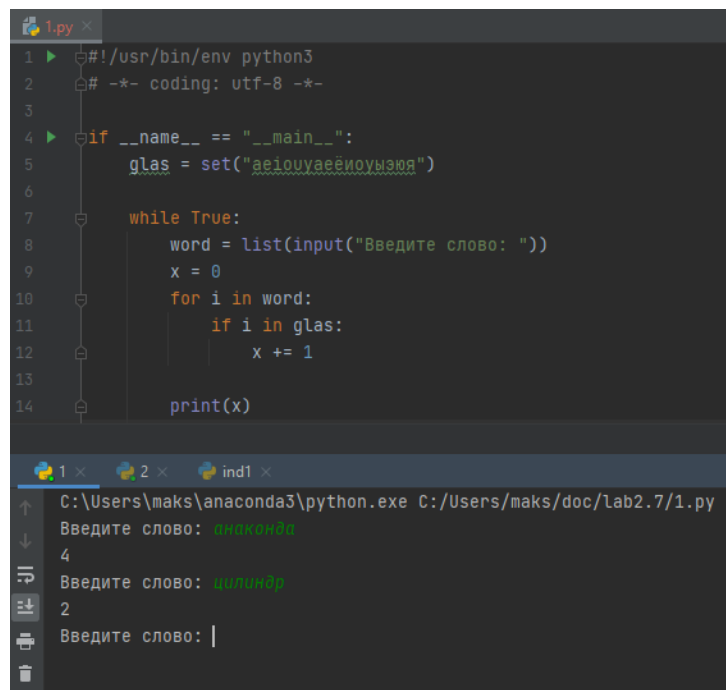
```
9 b = {"d", "f", "g", "o", "v", "y"}
10 c = {"d", "e", "j", "k"}
11 d = {"a", "b", "f", "g"}
12
13 x = (a.intersection(b)).union(c)
14 print(f"x = {x}")
15
16 # Найдем дополнения множеств
17 bn = u.difference(b)
18 cn = u.difference(c)
19
20 y = (a.difference(d)).union(cn.difference(bn))
21 print(f"y = {y}")
```

The bottom pane shows the execution output for the script:

```
C:\Users\maks\anaconda3\python.exe C:/Users/maks/doc/lab2.7/primer1.py
x = {'e', 'o', 'd', 'k', 'j'}
y = {'y', 'h', 'v', 'f', 'o', 'g', 'c'}
Process finished with exit code 0
```

Рисунок 1 – Пример

3. Выполнены задания из методики, рис. 2-3

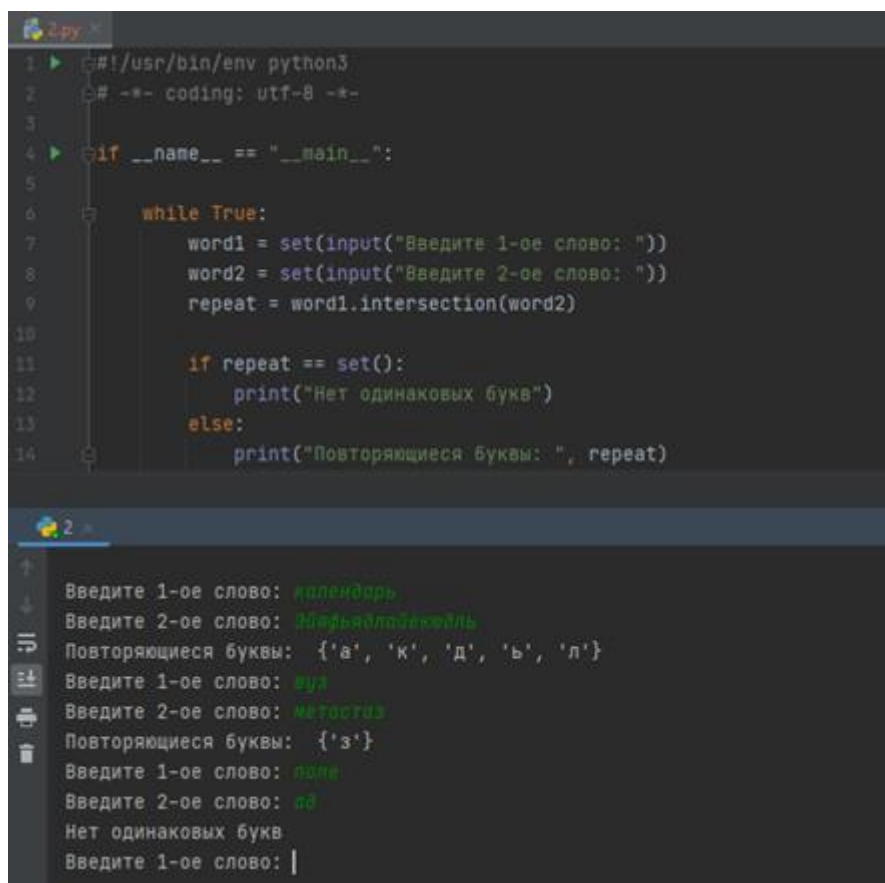


```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  if __name__ == "__main__":
5      glas = set("aeiouyаеёиоуыэюя")
6
7      while True:
8          word = list(input("Введите слово: "))
9          x = 0
10         for i in word:
11             if i in glas:
12                 x += 1
13
14         print(x)
```

The terminal output shows the script being executed with the following inputs and results:

```
C:\Users\maks\anaconda3\python.exe C:/Users/maks/doc/lab2.7/1.py
Введите слово: анаконда
4
Введите слово: цилиндр
2
Введите слово: |
```

Рисунок 2 – Задание 1



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  if __name__ == "__main__":
5
6      while True:
7          word1 = set(input("Введите 1-ое слово: "))
8          word2 = set(input("Введите 2-ое слово: "))
9          repeat = word1.intersection(word2)
10
11         if repeat == set():
12             print("Нет одинаковых букв")
13         else:
14             print("Повторяющиеся буквы: ", repeat)
```

The terminal output shows the script being executed with the following inputs and results:

```
Введите 1-ое слово: календарь
Введите 2-ое слово: январьдоложенье
Повторяющиеся буквы: {'а', 'к', 'д', 'ь', 'л'}
Введите 1-ое слово: вул
Введите 2-ое слово: метастаз
Повторяющиеся буквы: {'з'}
Введите 1-ое слово: лане
Введите 2-ое слово: ад
Нет одинаковых букв
Введите 1-ое слово: |
```

Рисунок 3 – Задание 2

4. Выполнено индивидуальное задание варианта 6, рис. 4



```
lab2.py
E: > lab2.py
1  if __name__ == "__main__":
2      # Определим универсальное множество
3      u = set("abcdefghijklmnopqrstuvwxyz")
4      a = {"b", "e", "g", "h", "k", "s"}
5      b = {"c", "g", "p", "q"}
6      c = {"k", "l", "w", "x"}
7      d = {"e", "j", "o", "p", "q", "u", "v"}
8      x = (a.union(b)).intersection(c)
9      print(f"x = {x}")
10     # Найдем дополнения множеств
11     an = u.difference(a)
12     y = (an.intersection(d)).union(c.difference(b))
13     print(f"y = {y}")
14
```

Рисунок 4 – Индивидуальное задание

Контрольные вопросы:

1. Что такое множества в языке Python?

Множеством в языке программирования Python называется неупорядоченная совокупность уникальных значений. В качестве элементов этого набора данных могут выступать любые неизменяемые объекты, такие как числа, символы, строки. В отличие от массивов и списков, порядок следования значений не учитывается при обработке его содержимого.

2. Как осуществляется создание множеств в Python?

Для создания множества нужно просто присвоить переменной последовательность значений, выделив их фигурными скобками:

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

Существует и другой способ создания множеств, который подразумевает использование вызова `set`. Аргументом этой функции может быть набор неких данных или даже строка с текстом.

3. Как проверить присутствие/отсутствие элемента в множестве?

Для этого используется `in`:

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

```
print(2 in a)
```

```
True
```

4. Как выполнить перебор элементов множества?

Перебор всех элементов выполняется циклом `for`:

```
for a in {0, 1, 2}:
```

```
    print(a)
```

5. Что такое set comprehension?

Для создания множества в Python можно воспользоваться генератором Set Comprehensions, позволяющим 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`. Аргументом в данном случае будет добавляемый элемент последовательности.

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.

Чтобы полностью убрать все элементы, надо использовать метод `clear`, не принимающий аргументов. Если вывести содержимое после этой операции, на экране появится только его название

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:

```
a = {0, 1, 2, 3}  
b = {4, 3, 2, 1}
```

```
c = a.union(b)
```

Чтобы найти общие элементы для двух разных множеств, следует применить функцию `intersection`, принимающую в качестве аргумента один из наборов данных:

```
c = a.intersection(b)
```

Чтобы вычислить разность для двух разных множеств, необходимо воспользоваться методом `difference`. Функция позволяет найти элементы, уникальные для второго набора данных, которых в нем нет:

```
c = a.difference(b)
```

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

Для определения подмножеств и надмножеств существуют специальные функции, возвращающие `True` или `False` в зависимости от результата выполнения. Для определения принадлежности элемента к множеству используется оператор `in`. Для того, чтобы проверить, что элемент не входит в множество используется оператор `not in`.

10. Каково назначение множеств `frozenset`?

Множество, содержащее 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?

Для преобразования множества в строку используется конкатенация текстовых значений, которую обеспечивает функция `join`.

Чтобы получить из множества словарь, следует передать функции `dict` набор из нескольких пар значений, в каждом из которых будет находиться ключ.

По аналогии с предыдущими преобразованиями можно получить список неких объектов. На этот раз используется вызов `list`, получающий в качестве аргумента множество `a`.

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