

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №2.7
дисциплины «Программирование на Python»
Вариант ____

Выполнил:
Иващенко Олег Андреевич
2 курс, группа ИВТ-б-о-22-1,
09.03.02 «Информационные и
вычислительные машины»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем»

(подпись)

Руководитель практики:
Воронкин Роман Александрович,
доцент кафедры инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: «Работа с множествами в языке Python»

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

Порядок выполнения работы

Таблица 1 – Код программы example.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

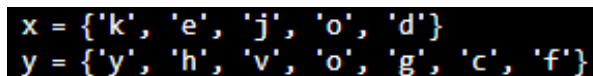
if __name__ == "__main__":
    # Определим универсальное множество
    u = set("abcdefghijklmnopqrstuvwxyz")

    a = {"b", "c", "h", "o"}
    b = {"d", "f", "g", "o", "v", "y"}
    c = {"d", "e", "j", "k"}
    d = {"a", "b", "f", "g"}

    x = (a.intersection(b)).union(c)
    print(f"x = {x}")

    # Найдем дополнения множеств
    bn = u.difference(b)
    cn = u.difference(c)

    y = (a.difference(d)).union(cn.difference(bn))
    print(f"y = {y}")
```



```
x = {'k', 'e', 'j', 'o', 'd'}
y = {'y', 'h', 'v', 'o', 'g', 'c', 'f'}
```

Рисунок 1 – Вывод программы example.py

Задание 1. Подсчитать количество гласных в строке, введенной с клавиатуры с использованием множеств.

Таблица 2 – Код программы individual_1.py

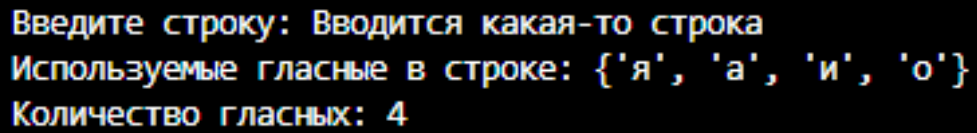
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == "__main__":
    string = list(input("Введите строку: "))
```

```

u = {"a", "y", "o", "ы", "э", "я", "ю", "ё", "и", "е"}
x = set(u.intersection(string))
print(f"Используемые гласные в строке: {x}")
print(f"Количество гласных: {len(x)}")

```



Введите строку: Вводится какая-то строка
Используемые гласные в строке: {'я', 'а', 'и', 'о'}
Количество гласных: 4

Рисунок 2 – Вывод программы individual_1.py

Задание 2. Определите общие символы в двух строках, введенных с клавиатуры.

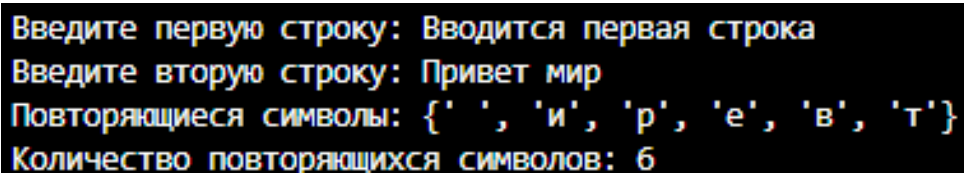
Таблица 3 – Код программы individual_2.py

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == "__main__":
    first_string = set(list(input("Введите первую строку: ")))
    second_string = list(input("Введите вторую строку: "))
    x = first_string.intersection(second_string)
    print(f"Повторяющиеся символы: {x}")
    print(f"Количество повторяющихся символов: {len(x)}")

```



Введите первую строку: Вводится первая строка
Введите вторую строку: Привет мир
Повторяющиеся символы: {' ', 'и', 'р', 'е', 'в', 'т'}
Количество повторяющихся символов: 6

Рисунок 3 – Вывод программы individual_2.py

Индивидуальное задание. Определить результат выполнения операций над множествами. Считать элементы множества строками.

$$A = \{a, b, g, k, t, p\}; B = \{b, e, f, l, r\}; C = \{k, l, w, x\}; D = \{e, j, o, p, q, u, v\};$$

$$X = (A / B) \cap (C \cup D); Y = (\neg A \cap \neg B) / (C \cup D)$$

Таблица 4 – Код программы individual_3.py

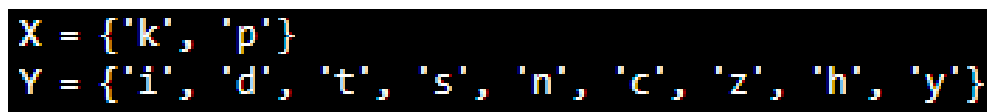
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

if __name__ == "__main__":
    u = set("abcdefghijklmnopqrstuvwxyz")
    a = {"a", "b", "g", "k", "m", "p"}
    b = {"b", "e", "f", "l", "r"}
    c = {"k", "l", "w", "x"}
    d = {"e", "j", "o", "p", "q", "u", "v"}

    ab = a.difference(b)
    cd = c.union(d)
    x = ab.intersection(cd)

    not_a = u.difference(a)
    not_b = u.difference(b)
    ab = not_a.intersection(not_b)
    y = ab.difference(cd)

    print(f"X = {x}")
    print(f"Y = {y}")
```



```
X = {'k', 'p'}
Y = {'i', 'd', 't', 's', 'n', 'c', 'z', 'h', 'y'}
```

Рисунок 4 – Вывод программы individual_3.py

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 и строки.

2. Как осуществляется создание множеств в Python?

Создать множество можно просто присвоив переменной последовательность значений, выделив их фигурными скобками.

3. Как проверить присутствие/отсутствие элемента в множестве?

Для проверки присутствия элемента в множестве можно использовать оператор `in`, например:

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

4. Как выполнить перебор элементов множества?

Перебор всех элементов множества можно осуществить с помощью цикла `for`, например:

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

5. Что такое set comprehension?

Set comprehension (множественное включение) – это способ определения множества в некоторых языках программирования, в том числе в языке Python. Этот метод позволяет создавать множества, указывая условия для его элементов. Например:

```
new_set = {x**2 for x in range(11)}
```

6. Как выполнить добавление элемента во множество?

Чтобы внести новое значение, потребуется вызвать метод `add`, аргументом которого 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.

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

Помимо различных манипуляций с элементами множеств существуют ещё и операции над ними, позволяющие одной строкой кода выполнять сложные преобразования:

- `A.union(B)` – объединение множества `A` с множеством `B`;
- `A.update(B)` – добавление всех элементов из множества `B` в множество `A`;
- `A.intersection(B)` – пересечение элементов множества `A` с элементами множества `B`;
- `A.difference(B)` – разность элементов `A` и элементов `B`;

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

Для определения подмножеств и надмножеств существуют специальные функции, возвращающие `False` или `True` в зависимости от результата выполнения. Для определения принадлежности элемента к множеству используют оператор `in` или `not in`.

Чтобы выяснить, является ли множество `A` подмножеством `B`, стоит попробовать вывести на экран результат выполнения метода `issubset`. Если не все элементы множества `A` присутствуют в `B`, функция вернёт `False`.

Чтобы узнать, является ли множество `A` надмножеством `B`, необходимо вызвать метод `isuperset`. Если все элементы множества `B` присутствуют в множестве `A`, то функция возвращает `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. Поскольку содержимое `frozenset` должно всегда оставаться статичным, перечень функций, с которыми такое множество может взаимодействовать, имеет ограничения.

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. Символ в кавычках выступает в качестве символа, разделяющего значения.

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`.

Выводы: В процессе выполнения лабораторной работы были приобретены навыки по работе со списками при написании программ с помощью языка программирования Python версии 3.x, были написаны 4 программы: пример из лабораторной работы, 2 задания и индивидуальная задача.