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 практической работе №2.7

Дисциплины «Программирование на Python»

Выполнил:

Пустяков Андрей Сергеевич

2 курс, группа ИВТ-б-о-22-1,

09.03.01 «Информатика и вычислительная техника (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____

(подпись)

Руководитель практики:

Воронкин Р. А. кандидат технических наук, доцент, доцент кафедры инфокоммуникаций

_____

(подпись)

Ставрополь, 2023 г.

Тема: Работа с множествами в языке Python.

Цель: приобрести навыки по работе с множествами при написании программ с помощью языка программирования Python версии 3.x.

Ход работы:

Создание репозитория, клонирование репозитория, организация репозитория согласно модели ветвления «git-flow» (рис. 1).

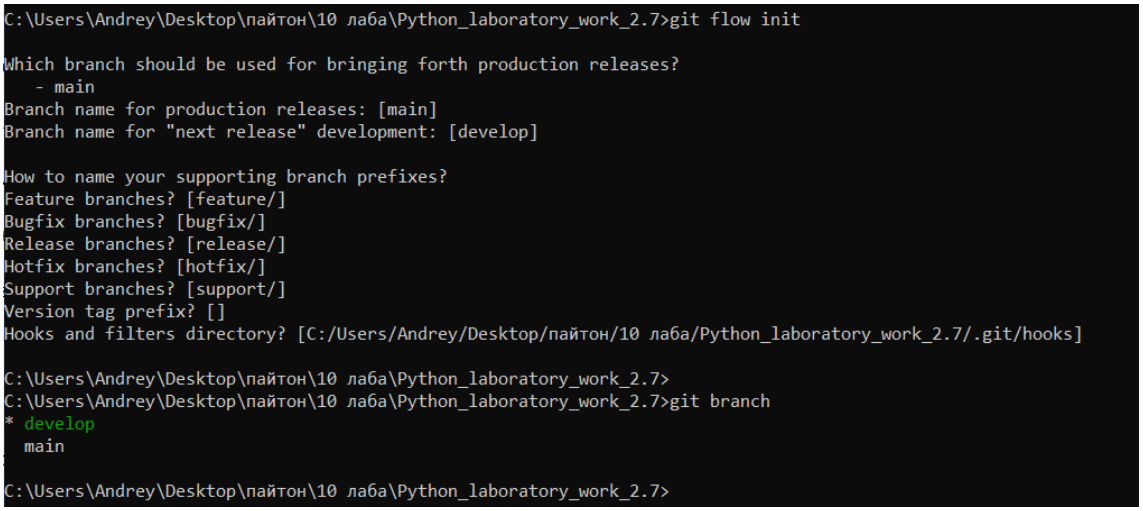

Рисунок 1 – Организация модели ветвления «git-flow».

Проработка примеров лабораторной работы:

Пример 1.

Необходимо определить результат выполнения операции над множествами:

$$A = \{b, c, h, o\}; \quad B = \{d, f, g, o, v, y\}; \quad C = \{d, e, j, k\}; \quad D = \{a, b, f, g\}; \quad X = (A \cap B) \cup C; \quad Y = (A/D) \cup (\bar{C}/\bar{B}).$$

Считать элементы множества строками.

Код программы примера №1 и результаты работы программы (рис. 2, 3).

```python
#!/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}")
```

Рисунок 2 – Код программы примера №1.

```
C:\Users\Andrey\AppData\Local\Programs\Python\Python39\pyth
x = {'j', 'k', 'o', 'e', 'd'}
y = {'y', 'g', 'v', 'f', 'c', 'h', 'o'}

Process finished with exit code 0
```

Рисунок 3 – Результаты работы программы примера №1.

Необходимо решить задачу: посчитать количество гласных в строке, введенной с клавиатуры с использованием множеств.

Код программы решения данной задачи и результаты работы программы (рис. 4, 5).

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

# Необходимо решить задачу: посчитать количество гласных в строке, введенной
# с клавиатуры с использованием множеств.

if __name__ == "__main__":
    letters_gl = {'о', 'е', 'ё', 'а', 'и', 'у', 'я', 'ы', 'ю', 'я'}
    words = input("Введите предложение: ")
    print("Количество гласных букв в этом предложении: ")
    count = 0
    for letter in words.lower():
        if letter in letters_gl:
            count += 1
    print(count)
```

Рисунок 4 – Код программы первой задачи.

```
C:\Users\Andrey\AppData\Local\Programs\Python\Python39\python.exe "C:\User
Введите предложение: Видимость мира делает войну еще опаснее. (Клавдиан).
Количество гласных букв в этом предложении:
19

Process finished with exit code 0
```

Рисунок 5 – Результаты работы программы первой задачи.

Необходимо решить задачу: необходимо определить общие символы в двух строках, введенных с клавиатуры.

Код программы решения задачи №2 и результаты работы программы (рис. 6, 7).

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

# Необходимо решить задачу: необходимо определить общие
# символы в двух строках, введенных с клавиатуры.

if __name__ == "__main__":
    first_sentence = input("Введите первое продложение: ")
    second_sentence = input("Введите второе предложение: ")
    symbols = set(second_sentence).intersection(set(first_sentence))
    print("Общие символы находящиеся в обоих предложениях: \n", ','.join(symbols))
```

Рисунок 6 – Код программы второй задачи.

```
C:\Users\Andrey\AppData\Local\Programs\Python\Python39\python.exe "C:\Users\Andrey
Введите первое продложение: Необходимо решить задачу: необходимо определить общие
Введите второе предложение: символы в двух строках, введенных с клавиатуры.
Общие символы находящиеся в обоих предложениях:
 е,н,д,у,и,о,м,р, ,л,а,т,х
```

Рисунок 7 – Результаты работы программы второй задачи.

Выполнение индивидуального задания:

Необходимо определить результат выполнения операции над множествами (Вариант 26):

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

$$A = \{a, b, c, d, e, r\}; \quad B = \{b, c, d, f, n, y\}; \quad C = \{b, c, h, k, l, s\}; \quad D = \{a, b, r, s, w, x\};$$

Считать элементы множества строками. Результаты необходимо проверить вручную.

Код программы решения индивидуального задания и результаты работы программы (рис. 8, 9).

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

# Необходимо определить результат выполнения операции над множествами.

if __name__ == "__main__":
    a = {'a', 'b', 'c', 'd', 'e', 'r'}
    b = {'b', 'c', 'd', 'f', 'n', 'y'}
    c = {'b', 'c', 'h', 'k', 'l', 's'}
    d = {'a', 'b', 'r', 's', 'w', 'x'}

    # Так как в выражении есть логическое 'не', то нужно создать универсум.
    u = {'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i',
         'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r',
         's', 't', 'u', 'v', 'w', 'x', 'y', 'z'}

    # Запишем решение данных операций над множествами.
    x = (a.union(d)).intersection(c)
    y = (a.difference(d)).union((u.difference(c)).difference(u.difference(b)))
    print("X = \n", x)
    print("Y = \n", y)
```

Рисунок 8 – Код программы индивидуального задания.

```
C:\Users\Andrey\AppData\Local\Programs\Python\Python39\python.exe
X =
 {'b', 'c', 's'}
Y =
 {'d', 'n', 'c', 'y', 'e', 'f'}
```

Рисунок 9 – Результаты работы программы индивидуального задания.

Ответы на контрольные вопросы:

1. Множество – это неупорядоченная совокупность уникальных значений. В качестве элементов этого набора данных могут выступать любые неизменяемые объекты (числа, символы, строки). В отличие от массивов и списков, порядок следования значений не учитывается при обработке его содержимого. Над одним, а также несколькими множествами можно выполнять ряд операций, благодаря функциям стандартной библиотеки языка программирования Python.

2. Множество можно создавать с помощью метода set(«элементы») или присваиванием переменной значений в фигурных скобках ({‘a’, 10, tuple, sss’}).

3. Для проверки наличия в множестве требуемого элемента, можно использовать оператор «in», возвращающий True, если в множестве есть элемент и False, если нет. Если же нужно проверить отсутствие, то можно использовать «not in».

4. Цикл «for» с методом «in» позволяет пройтись по всем элементам множества. Относительно индексов не получится пройти по множествам, ведь множества являются неупорядоченными.

5. Для создания множества можно в Python воспользоваться генератором, называющимся «set comprehension», и позволяющим заполнять списки, а также другие наборы данных с учетом неких условий. Пример - генерация множества a с циклом «for» для нескольких чисел: a = {i for i in [1, 2, 0, 1, 3, 2]}.

6. Метод «add()» позволяет добавить элемент в множество.

7. Для удаления элементов из множества есть несколько методов: «remove» (Удаление требуемого элемента. Если такого в множестве нет, появляется исключение), «discard» (Удаление элемента без генерации исключения), «pop» (Удаление элемента в множестве, стоящего в тот момент первым).

8. Существует несколько методов для работы над множествами: «union» (Складывание множеств, при этом создаётся новое множество), «update» (Складывание множеств, результат передаётся множеству, в которое добавлялись элементы), «intersection» (Пересечение множеств, результат передаётся новому множеству), «difference» (Разность множеств, результат (Элементы, не хранящиеся в вычитаемом множестве), передаётся новому множеству).

9. Метод «isupperset()» позволяет определить, является ли множество надмножеством другого множества. Метод «issubset()», в свою очередь, позволяет определить, является ли множество подмножеством.

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

11. Для преобразования множества в строку, можно использовать функцию «join», а для преобразования множества в словарь, можно использовать функцию «dict()», однако при этом во множестве должны быть объекты, содержащие два значения, которые соответственно будут определены как ключ и значение. Для преобразования в список, можно использовать «list()».

Вывод: в ходе выполнения лабораторной работы была изучена работа с множествами в языке программирования Python. Были изучены способы создания множеств и применение множеств. Также были изучены способы изменения множеств, способы преобразования множеств в другие типы данных и обратно. Были более подробно изучены методы, позволяющие производить операции над множествами. Были изучены способы определения отношения над множествами.