# МИНИCTEPCTBO НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙ- СКОЙ ФЕДЕРАЦИИ

**Федеральное государственное автономное образовательное учреждение высшего образования**

# «СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

**Кафедра инфокоммуникаций**

# Основы кроссплатформенного программирования Отчет по лабораторной работе №2.7

Тема: «Работа с множествами в языке Python»

|  |
| --- |
| Выполнил студент группы  ИВТ-б-о-21-1 |
| Артемьев А.В « » 20 г. |
| Подпись студента |
| Работа защищена « » 20 г. |
| Проверил доцент  Кафедры инфокоммуникаций, старший преподаватель  Воронкин Р.А.  (подпись) |

Ставрополь 2022

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

# Ход работы:

1. **Создал репозиторий в GitHub,**  выбрал лицензию MIT, клонировал его на компьютер и организовал в соответствии с моделью ветвления git-flow.

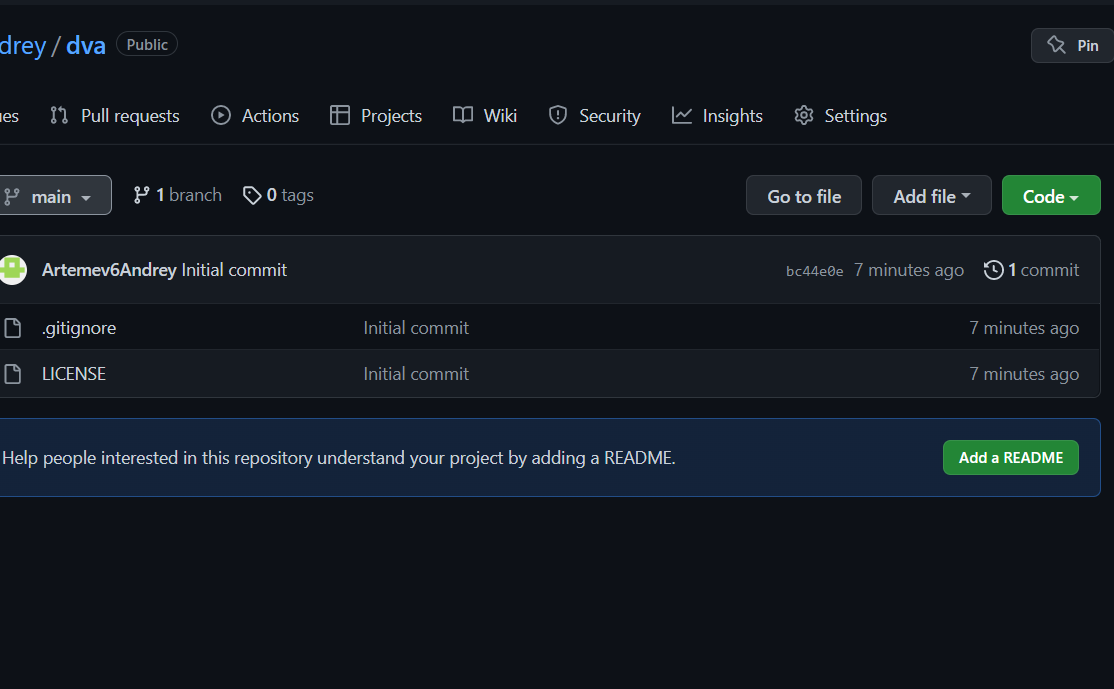


Рисунок 1.1 – Созданный репозиторий

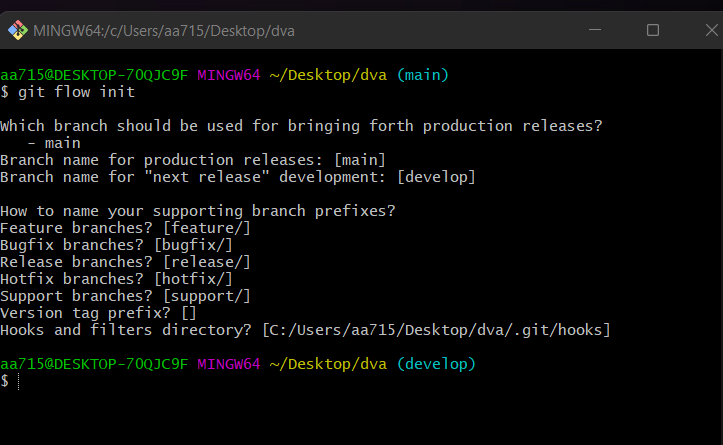


Рисунок 1.2 – Организация репозитория в соответствии с моделью ветвления git-flow

1. **Создал проект** Pycharm в папке репозитория, проработал примеры ЛР.

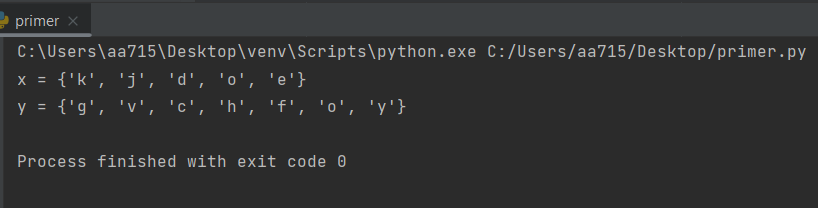


Рисунок 2.1 – Результат выполнения примера №1

1. **Решить задачу №1:** подсчитайте количество гласных в строке, вве- денной с клавиатуры с использованием множеств.

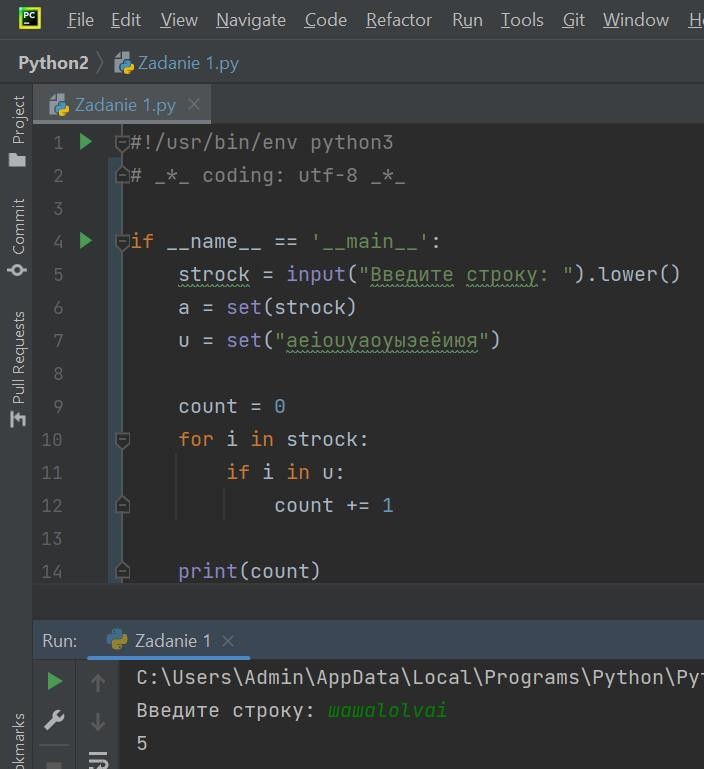


Рисунок 3.1 – Выполненное задание

# Решите задачу №2: определите общие символы в двух строках, введенных с клавиатуры.

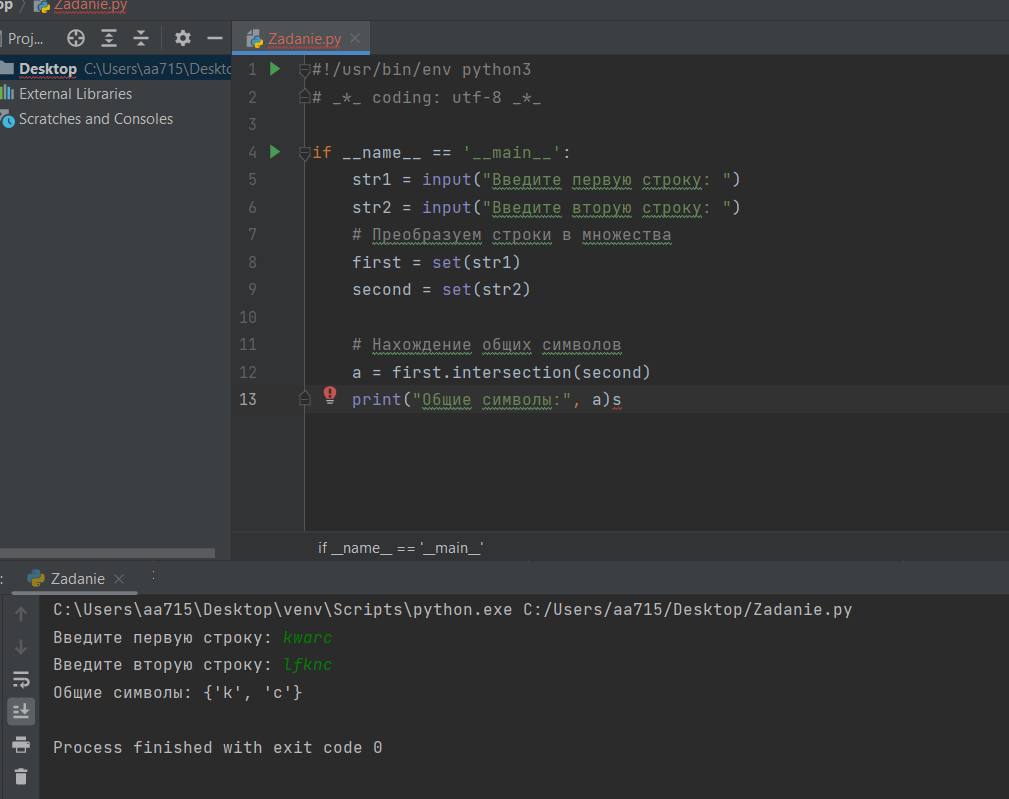
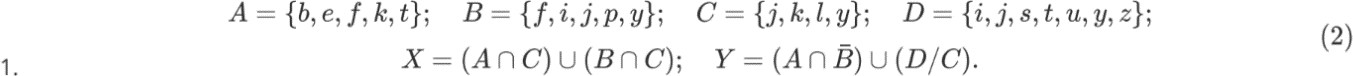


Рисунок 4.1 – Выполненное задание

1. **Индивидуальное задание В – 1.** Определить результат выполнения операций над множествами.

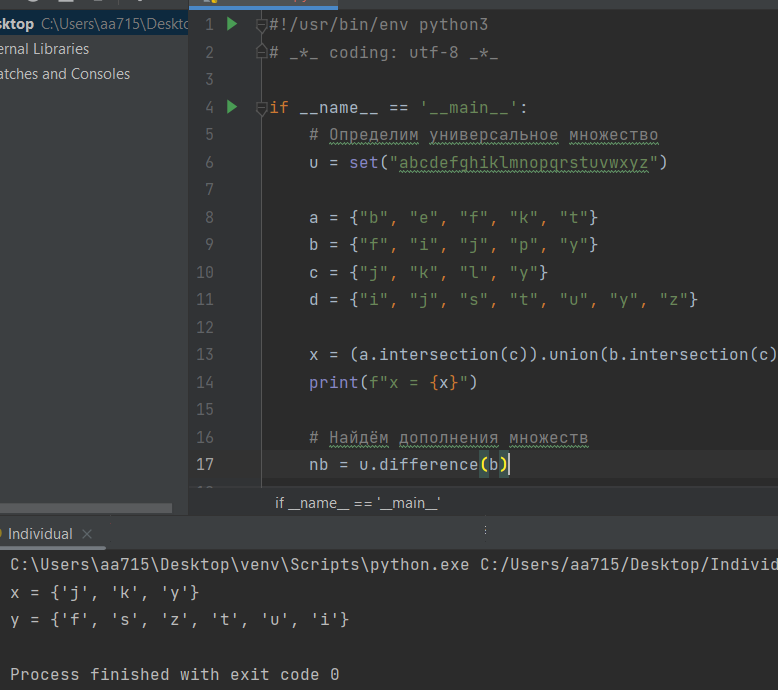


Рисунок 5.2 – Индивидуальное задание

**Вывод:** в результате выполнения лабораторной работы были приобре- тены навыки для работы с множествами при написании программ с помощью языка программирования Python версии 3.x.

# Ответы на контрольные вопросы:

1. **Что такое множества в языке Python?**

Множеством в языке программирования Python называется неупорядо- ченная 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. Над одним, а также

несколькими множествами можно выполнять ряд операций, благодаря функ- циям стандартной библиотеки языка программирования Python.

# Как осуществляется создание множеств в Python?

Создать можно, просто присвоив переменной последовательность зна- чений, выделив их фигурными скобками. Следующий пример показывает код, в котором создается множество целых чисел под названием a, после функция print выводит на экран его содержимое.

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

print(a)

{0, 1, 2, 3}

Существует и другой способ создания множеств, который подразуме- вает использование вызова set. Аргументом этой функции может быть набор неких данных или даже строка с текстом, как это показано в следующем при- мере.

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

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

# 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

# Как выполнить перебор элементов множества?

Перебор всех элементов. for a in {0, 1, 2}:

print(a) 0

1

2

# Что такое set comprehension?

**Set Comprehensions – для создания множества можно в Python вос- пользоваться генератором, позволяющих заполнять списки, а также дру- гие наборы данных с учетом неких условий.** Следующий код демонстрирует генерацию множества a с циклом for для нескольких чисел:

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

{0, 1, 2, 3}

# Как выполнить добавление элемента во множество?

**Чтобы внести новые значения, потребуется вызывать метод add.** Аргументом в данном случае будет добавляемый элемент последовательно- сти. В примере кода на Python добавим в множество элемент со значением 4.

a = {0, 1, 2, 3}

a.add(4) print(a)

{0, 1, 2, 3, 4}

# Как выполнить удаление одного или всех элементов множества?

**Для удаления элементов из множества используются следующие функции в Python** (кроме очистки, которая будет рассмотрена ниже):

* + **remove — удаление элемента с генерацией исключения** в слу- чае, 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.

Избавиться от лишних значений в наборе данных с помощью remove. В качестве входного параметра здесь выступает элемент, который нужно уда- лить (в примере удалим число со значением 3).

# Как выполняются основные операции над множествами: объеди- нение, пересечение, разность?

**Объединение**

Чтобы объединить все элементы двух разных множеств, стоит восполь- зоваться методом union на одном из объектов. Следующий пример демонстри- рует работу данной функции, где создается последовательность чисел под именем c.

a = {0, 1, 2, 3}

b = {4, 3, 2, 1}

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

{0, 1, 2, 3, 4}

# Пересечение

Чтобы найти общие элементы для двух разных множеств, следует при- менить функцию intersection, принимающую в качестве аргумента один из наборов данных. Код, приведенный ниже, создает новую последовательность чисел из пересечения двух множеств в Python 3.

# Разность

Чтобы вычислить разность для двух разных множеств, необходимо вос- пользоваться методом difference. Функция позволяет найти элементы, уни- кальные для второго набора данных, которых в нем нет. Следующий код де- монстрирует эту операцию.

a = {0, 1, 2, 3}

b = {4, 3, 2, 1}

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

{0}

# Как определить, что некоторое множество является надмноже- ством или подмножеством другого множества?

Чтобы выяснить, является ли множество a подмножествомb, стоит по- пробовать вывести на экран результат выполнения метода issubset, как в сле- дующем примере. Так как не все элементы набора чисел a присутствуют в b, функция вернет False.

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

b = {3, 2, 1}

print(a.issubset(b)) False

# Определение надмножества

Чтобы узнать, является ли множество a надмножеством b, необходимо вызвать метод issuperset и вывести результат его работы на экран. Поскольку все элементы набора чисел b присутствуют в a, функция возвращает True.

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

b = {3, 2, 1}

print(a.issuperset(b)) True

# Каково назначение множеств frozenset?

**Тип frozenset – множество, содержимое которого не поддается изме- нению имеет тип frozenset.** Значения из этого набора нельзя удалить, как и добавить новые. В следующем примере демонстрируется создание при по- мощи стандартной функции.

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

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

Поскольку содержимое frozenset должно всегда оставаться статичным, перечень функций, с которыми такое множество может взаимодействовать, имеет ограничения.

# Как осуществляется преобразование множеств в строку, список, словарь?

**Строка**

Для преобразования множества в строку используется конкатенация тек- стовых значений, которую обеспечивает функция join. В этом случае ее аргу- ментом 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 объ- екта, а type отображает его тип.

a = {('a', 2), ('b', 4)}

b = dict(a) print(b) print(type(b))

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

<class 'dict'>

Следует отметить, что каждый элемент для такого преобразования — кортеж состоящий из двух значений:

1. ключ будущего словаря;
2. значение, соответствующее ключу.

# Список

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'>