

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

Кафедра инфокоммуникаций

Языки программирования

Отчет по лабораторной работе № 1

«Работа с множествами в языке Python».

Выполнил студент группы

ИТС-б-о-20-1 (1)

Рудаков В.Б. « » _____ 20__ г.

Подпись студента _____

Работа защищена « » _____ 20__ г.

Проверил доцент

Кафедры инфокоммуникаций, кандидат
технических наук

Воронкин Р.А.

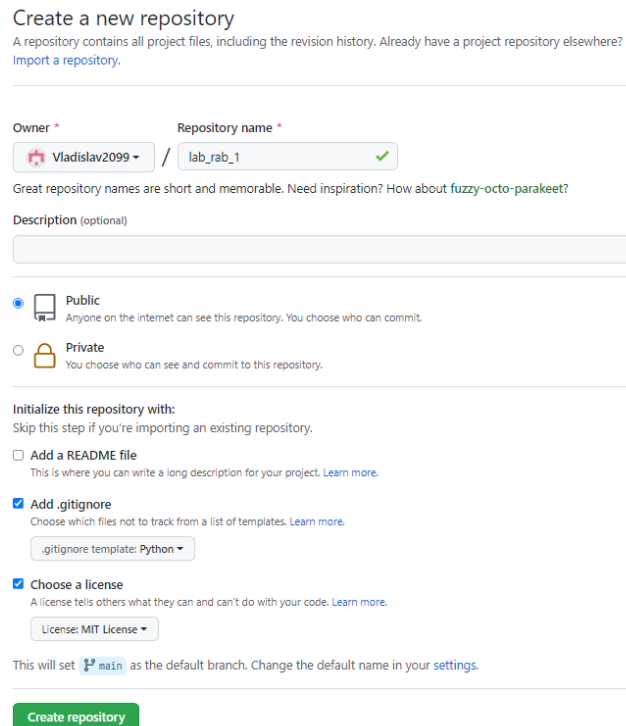
(подпись)

Ставрополь, 2021

Тема работы: Работа с множествами в языке Python.

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

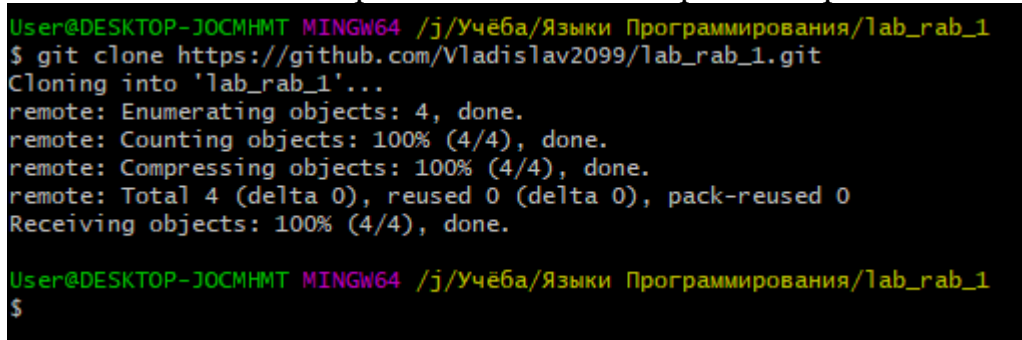
1. Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



The screenshot shows the GitHub 'Create a new repository' page. At the top, it says 'Create a new repository' and provides a brief explanation. Below this, there are two input fields: 'Owner' with a dropdown menu showing 'Vladislav2099' and 'Repository name' with a text input 'lab_rab_1' and a green checkmark. A note below these fields says 'Great repository names are short and memorable. Need inspiration? How about fuzzy-octo-parakeet?'. There is a 'Description (optional)' text area. Under the 'Visibility' section, the 'Public' radio button is selected, with a sub-note 'Anyone on the internet can see this repository. You choose who can commit.' The 'Private' option is also visible. In the 'Initialize this repository with:' section, there are three checkboxes: 'Add a README file' (unchecked), 'Add .gitignore' (checked), and 'Choose a license' (checked). The '.gitignore' dropdown is set to 'Python'. The 'Choose a license' dropdown is set to 'MIT License'. At the bottom, it says 'This will set main as the default branch. Change the default name in your settings.' and a green 'Create repository' button.

Рисунок 1. Создание репозитория.

2. Выполните клонирование созданного репозитория.

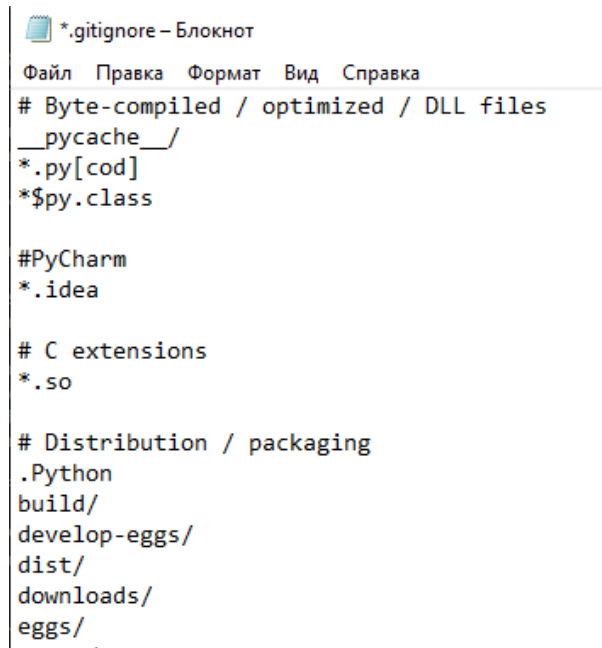


```
User@DESKTOP-JOCMHT MINGW64 /j/Учёба/Языки Программирования/lab_rab_1
$ git clone https://github.com/Vladislav2099/lab_rab_1.git
Cloning into 'lab_rab_1'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.

User@DESKTOP-JOCMHT MINGW64 /j/Учёба/Языки Программирования/lab_rab_1
$
```

Рисунок 2. Клонирование репозитория.

3. Дополните файл .gitignore необходимыми правилами для работы с IDE PyCharm.



```
*.gitignore – Блокнот
Файл  Правка  Формат  Вид  Справка
# Byte-compiled / optimized / DLL files
__pycache__/
*.py[cod]
*$py.class

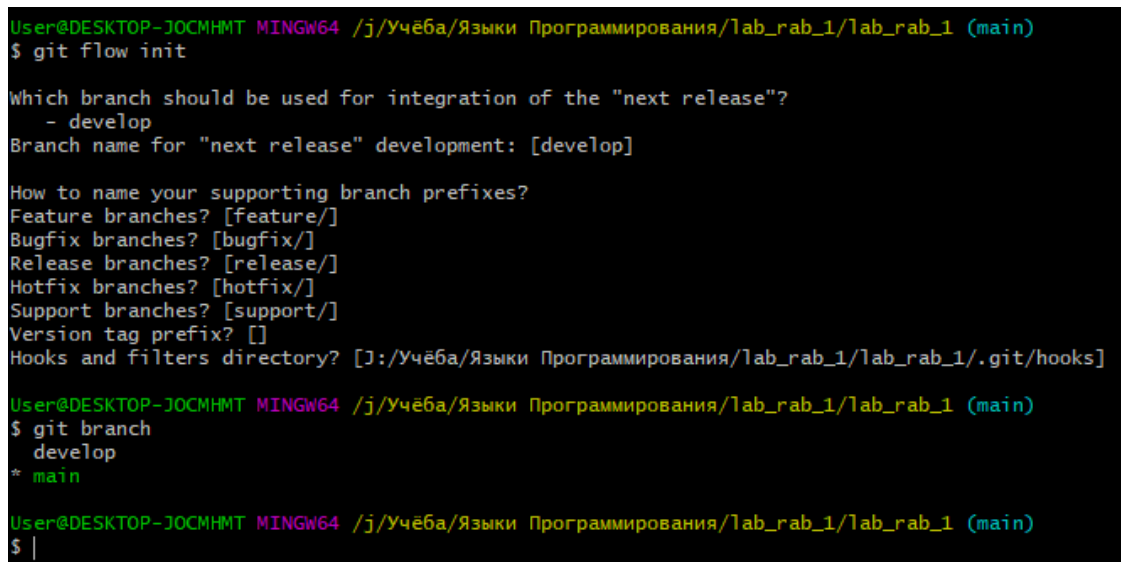
#PyCharm
*.idea

# C extensions
*.so

# Distribution / packaging
.Python
build/
develop-eggs/
dist/
downloads/
eggs/
```

Рисунок 3. Редактирование файла *.gitignore*.

4. Организуйте свой репозиторий в соответствии с моделью ветвления git-flow.



```
User@DESKTOP-JOCMHMT MINGW64 /j/Учеба/Языки Программирования/lab_rab_1/lab_rab_1 (main)
$ git flow init

Which branch should be used for integration of the "next release"?
- develop
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [j:/Учеба/Языки Программирования/lab_rab_1/lab_rab_1/.git/hooks]

User@DESKTOP-JOCMHMT MINGW64 /j/Учеба/Языки Программирования/lab_rab_1/lab_rab_1 (main)
$ git branch
  develop
* main

User@DESKTOP-JOCMHMT MINGW64 /j/Учеба/Языки Программирования/lab_rab_1/lab_rab_1 (main)
$ |
```

Рисунок 4. Модель ветвления *got-flow*.

5. Создайте проект PyCharm в папке репозитория.

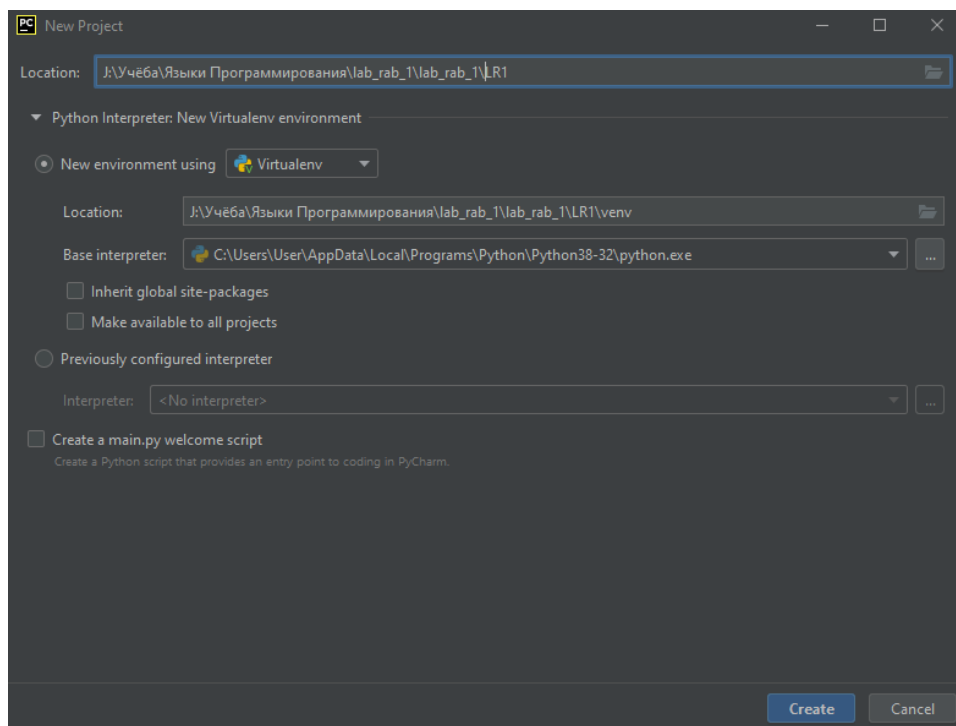


Рисунок 5. Создание проекта.

6. Проработайте примеры лабораторной работы. Создайте для них отдельный модуль языка Python. Зафиксируйте изменения в репозитории.

```

1  a = {1, 2, 0, 1, 3, 2}
2  print(a)
3
4
5  a = set('data')
6  print(a)
7
8
9  a = {'set', 'str', 'dict', 'list'}
10 b = ','.join(a)
11 print(b)
12 print(type(b))
13
14 a = {('a', 2), ('b', 4)}
15 b = dict(a)
16 print(b)
17 print(type(b))
18
19 a = {1, 2, 0, 1, 3, 2}
20 b = list(a)
21 print(b)
22 print(type(b))
23
24 a = {0, 1, 12, 'b', 'ab', 3, 2, 'a'}
25 print(a)
26
27 a = {0, 1, 12, 3, 2}
28 print(a)
29
30 a = {0, 1, 12, 3, 2}

```

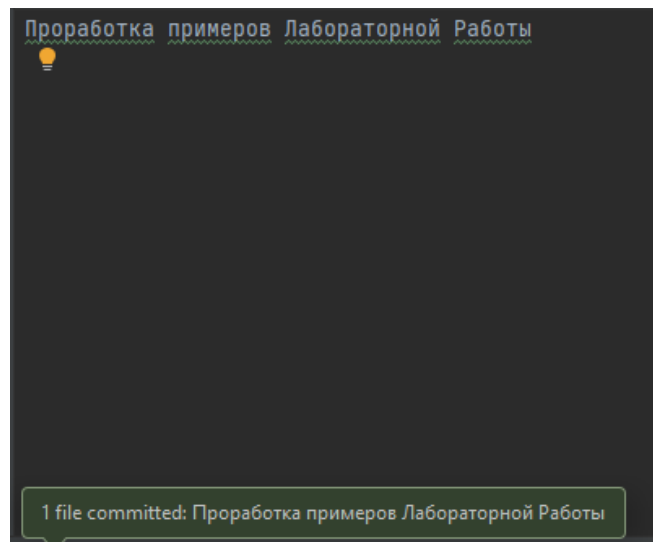


Рисунок 6. Примеры из работы.

7. Решите задачу: подсчитайте количество гласных в строке, введенной с клавиатуры с использованием множеств.

A screenshot of a Python script and its execution output. The script is named "Задача 1.py" and contains the following code:

```
1 vse = set("абвгдеёжзийклмнопрстуфхцчшщъыьэюя")
2 a = {"а", "о", "э", "е", "и", "ы", "у", "ё", "ю", "я"}
3 u1 = vse - a
4 c = input()
5 b = set(c)
6 u = b - u1
7 print(len(u))
```

The output shows the input string "Наша Таня громко плачет" and the result 8, indicating the number of vowels. The process finished with exit code 0.

```
Run: Задача 1
"J:\Учёба\Языки Программирования\lab_rab_1\lab_rab_1\LR1\
Наша Таня громко плачет
8
Process finished with exit code 0
```

Рисунок 7. Решение задачи.

8. Зафиксируйте сделанные изменения в репозитории.

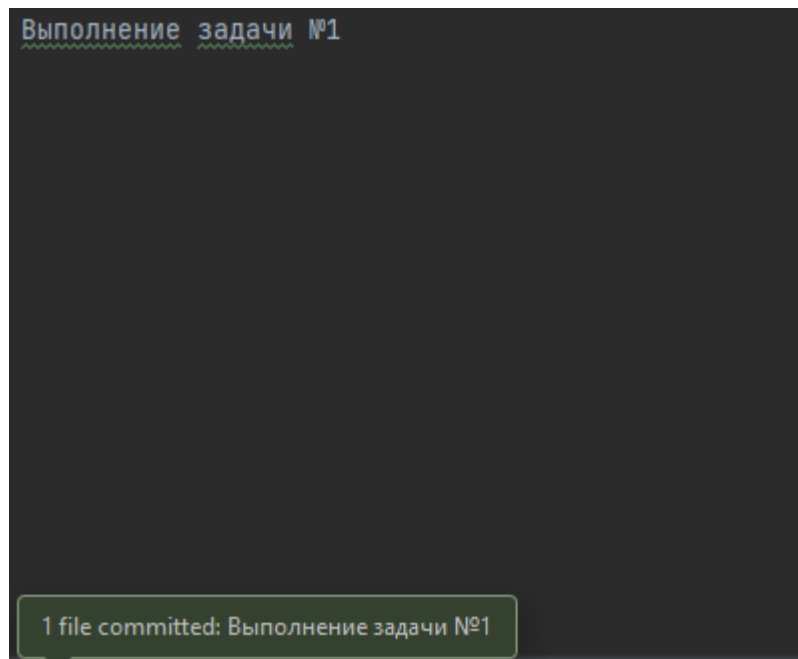


Рисунок 8. Фиксирование изменений.

9. Решите задачу: определите общие символы в двух строках, введенных с клавиатуры.

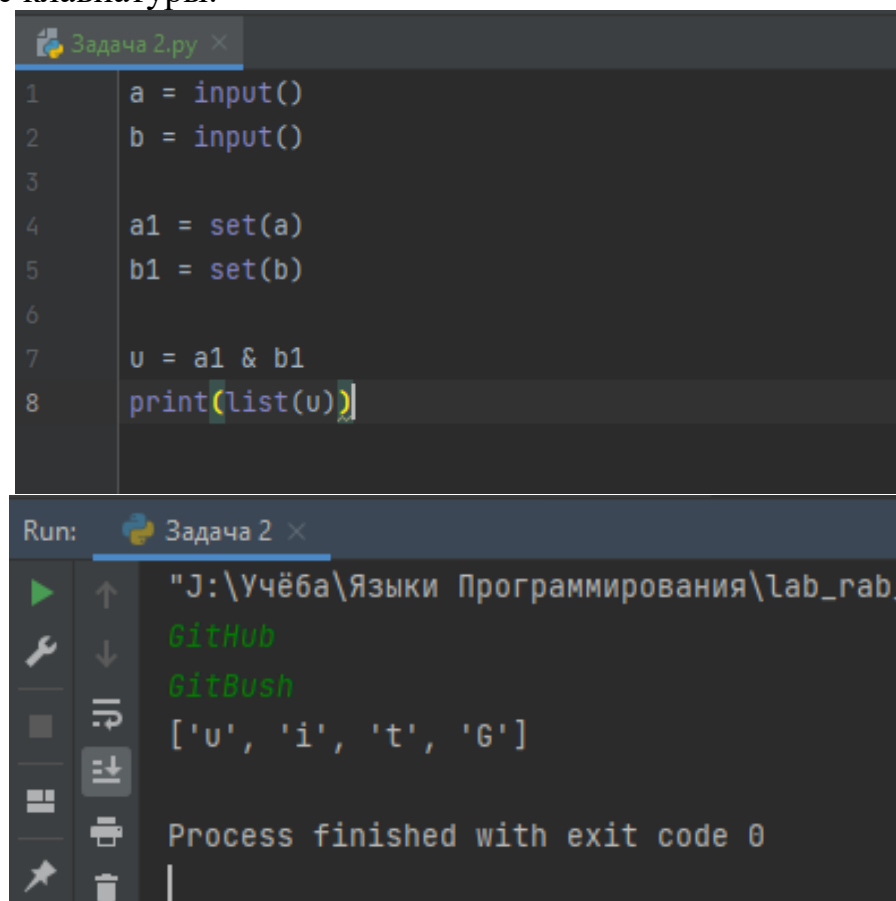


Рисунок 9. Решение задачи.

10. Зафиксируйте сделанные изменения в репозитории.

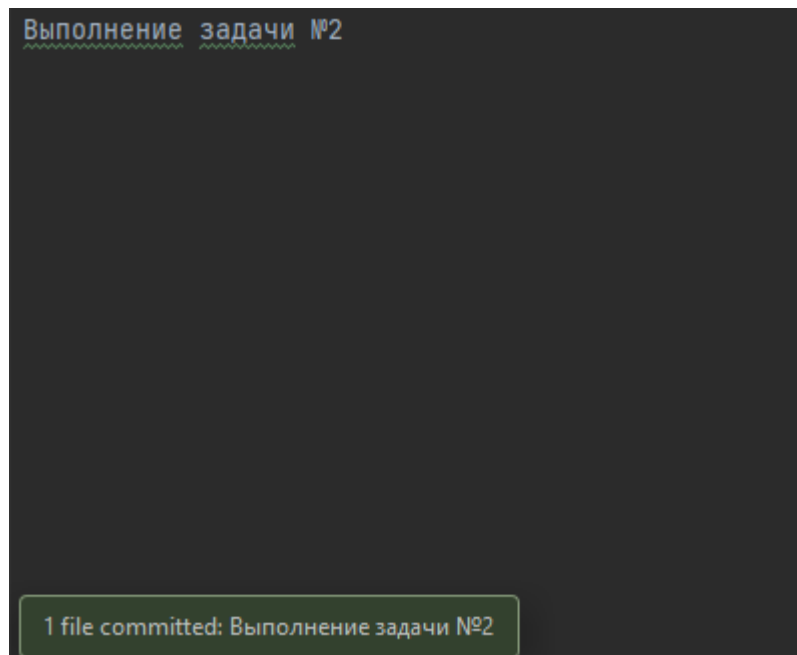


Рисунок 10. Фиксирование изменений.

11. Добавьте отчет по лабораторной работе в *формате PDF* в папку *doc* репозитория. Зафиксируйте изменения.

Рисунок 11. Добавление отчёта в репозиторий.

12. Выполните слияние ветки для разработки с веткой *master/main*.

```
User@DESKTOP-ЮСМНМТ MINGW64 /j/Учёба/Языки Программирования/lab_rab_1/lab_rab_1 (main)
$ git checkout develop
Switched to branch 'develop'

User@DESKTOP-ЮСМНМТ MINGW64 /j/Учёба/Языки Программирования/lab_rab_1/lab_rab_1 (develop)
$ git merge main
Updating a10ebb8..52a9522
Fast-forward
 LR1/Examples.py | 92 +++++
...7\320\260\320\264\320\260\321\207\320\260 1.py" | 7 ++
...7\320\260\320\264\320\260\321\207\320\260 2.py" | 8 ++
3 files changed, 107 insertions(+)
create mode 100644 LR1/Examples.py
create mode 100644 "LR1/\320\227\320\260\320\264\320\260\321\207\320\260 1.py"
create mode 100644 "LR1/\320\227\320\260\320\264\320\260\321\207\320\260 2.py"

User@DESKTOP-ЮСМНМТ MINGW64 /j/Учёба/Языки Программирования/lab_rab_1/lab_rab_1 (develop)
$ |
```

Рисунок 12. Слияние веток.

13. Отправьте сделанные изменения на сервер GitHub.

```

User@DESKTOP-ЮСМНМТ MINGW64 /j/Учёба/Языки Программирования/lab_rab_1/lab_rab_1 (develop)
$ git push origin develop
Enumerating objects: 17, done.
Counting objects: 100% (17/17), done.
Delta compression using up to 2 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (15/15), 1.82 KiB | 169.00 KiB/s, done.
Total 15 (delta 5), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (5/5), completed with 1 local object.
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/Vladislav2099/lab_rab_1/pull/new/develop
remote:
To https://github.com/Vladislav2099/lab_rab_1.git
 * [new branch]      develop -> develop

```

Рисунок 13. Отправка изменений.

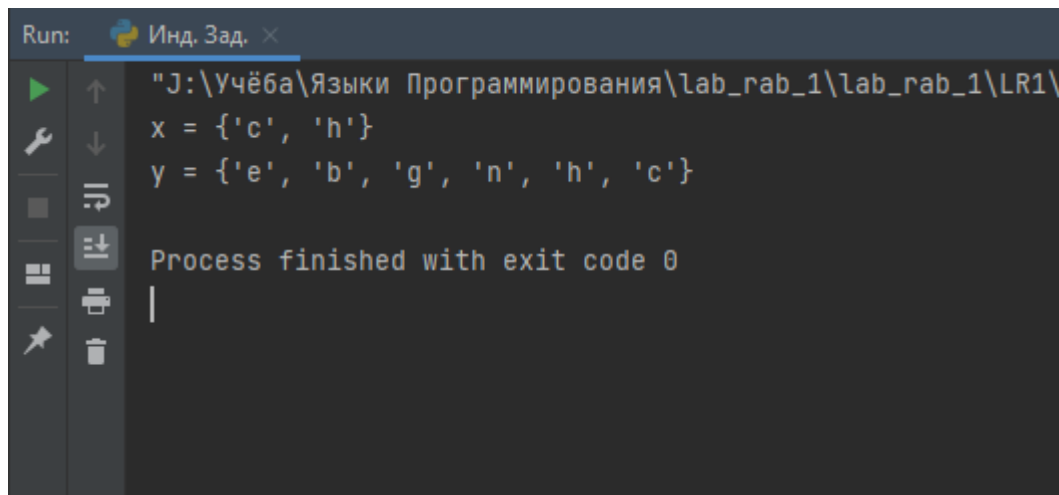
Индивидуальное задание:

5. $A = \{c, e, h, n\}; \quad B = \{e, f, k, n, x\}; \quad C = \{b, c, h, p, r, s\}; \quad D = \{b, e, g\};$ (6)
 $X = (A/B) \cap (C \cup D); \quad Y = (A \cap B) \cup (C/D).$

```

Инд. Зад..py x
J:\Учёба\Языки Программирования\lab_rab_1\lab_rab_1\LR1\Инд. Зад..py
3  ▶ if __name__ == "__main__":
4      # Определим универсальное множество
5      u = set("abcdefghijklmnopqrstuvwxyz")
6
7      a = {"c", "e", "h", "n"}
8      b = {"e", "f", "k", "n", "x"}
9      c = {"o", "p", "w"}
10     d = {"b", "e", "g"}
11     x = (a.difference(b)).difference(c.union(d))
12     print(f"x = {x}")
13
14     # Найдем дополнения множеств
15     dn = u.difference(d)
16     cn = u.difference(c)
17
18     y = (a.intersection(dn)).union(cn.difference(dn))
19     print(f"y = {y}")
20

```

```
Run: Инд. Зад. x
"J:\Учёба\Языки Программирования\lab_rab_1\lab_rab_1\LR1\
x = {'c', 'h'}
y = {'e', 'b', 'g', 'n', 'h', 'c'}

Process finished with exit code 0
```

Рисунок 14. Индивидуальное задание.

КОНТРОЛЬНЫЕ ВОПРОСЫ:

1. Что такое множества в языке Python?

Множеством в языке программирования Python называется неупорядоченная совокупность уникальных значений.

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 этого используется `in`.

4. Как выполнить перебор элементов множества?

Для этого используется `for ... in`.

5. Что такое *set comprehension*?

Генератор, позволяющий заполнять списки, а также другие наборы данных с учетом неких условий.

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.

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.

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

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

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

Чтобы выяснить, является ли множество **a** подмножеством **b**, стоит попробовать вывести на экран результат выполнения метода *issubset*.

Чтобы узнать, является ли множество **a** надмножеством **b**, необходимо вызвать метод *issuperset* и вывести результат его работы на экран.

10. Каково назначение множеств *frozenset*?

Множество, содержимое 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*. В этом случае ее

аргументом является набор данных в виде нескольких строк. Запятая 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. Функция *print* демонстрирует на экране содержимое полученного объекта, а *type* отображает его тип.

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.