

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

**Кафедра  
инфокоммуникаций  
Институт цифрового  
развития**

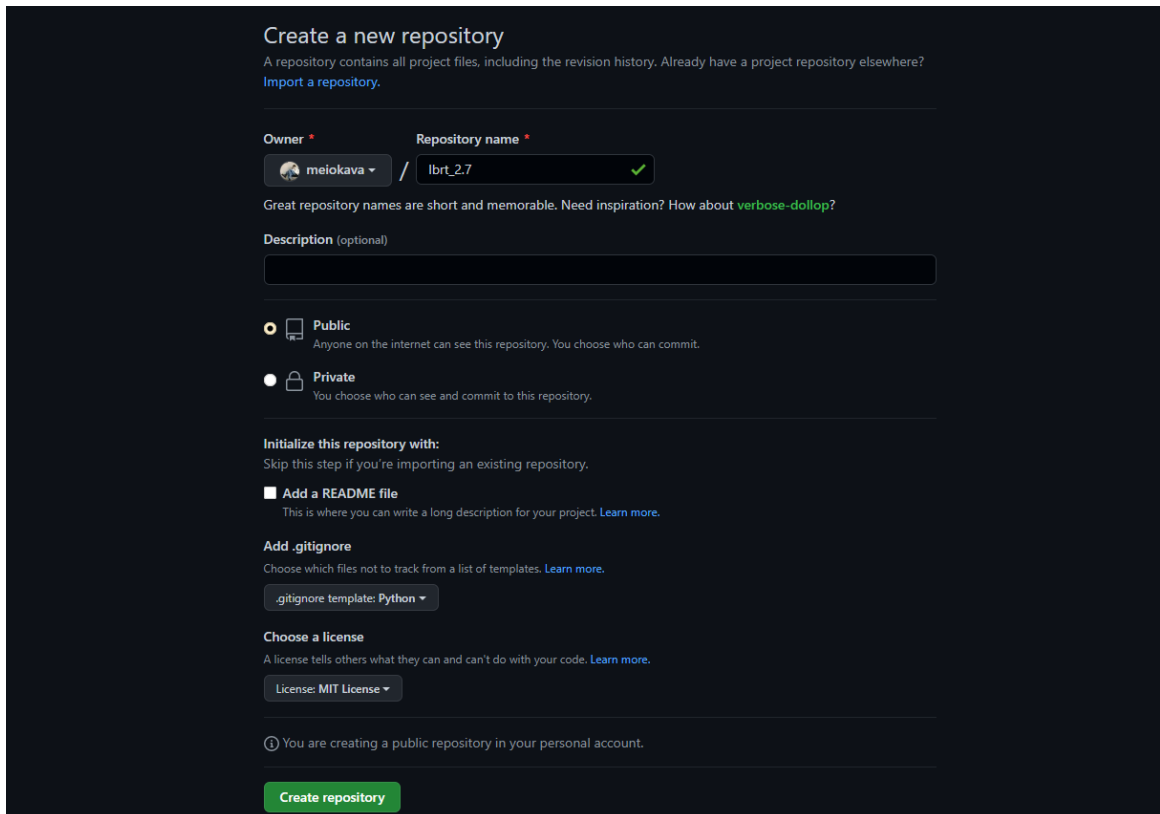
**ОТЧЁТ**  
**по лабораторной работе №2.7**  
Дисциплина: «Основы программной инженерии»  
Тема: «Работа с множествами в языке Python»

Выполнила:  
студентка 2 курса  
группы Пиж-б-о-21-1  
Джолдошова Мээрим  
Бекболотовна

Ставрополь 2022

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


1. Был создан репозиторий в Github в который были добавлены правила gitignore для работы IDE PyCharm, была выбрана лицензия MIT, сам репозиторий был клонирован на локальный сервер и был организован в соответствии с моделью ветвления git-flow.



Create a new repository


A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Owner <sup>\*</sup> Repository name <sup>\*</sup>

 meiokava / lbrt\_2.7 ✓

Great repository names are short and memorable. Need inspiration? How about [verbose-dollop](#)?

Description (optional)

☒  **Public**  
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**  
You choose who can see and commit to this repository.

**Initialize this repository with:**  
Skip this step if you're importing an existing repository.

☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more.](#)

**Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)

**Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)

① You are creating a public repository in your personal account.

[Create repository](#)

Рисунок 1 – Создание репозитория

```
c:\Users\мвидео>cd/d c:\lbrt_2.7
c:\lbrt_2.7>git clone https://github.com/meiokava/lbrt_2.7.git
Cloning into 'lbrt_2.7'...
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.
c:\lbrt_2.7>
```

Рисунок 2 – Клонирование репозитория

```

C:\Users\МВИДЕО>cd/d C:\lbrt_2.7\lbrt_2.7
C:\lbrt_2.7\lbrt_2.7>git flow init
which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
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? [C:\lbrt_2.7\lbrt_2.7\.git\hooks]

C:\lbrt_2.7\lbrt_2.7>git branch -a
* develop
  main
  remotes/origin/HEAD -> origin/main
  remotes/origin/main
C:\lbrt_2.7\lbrt_2.7>

```

Рисунок 3 – Организация модели ветвления Git flow

```

1  ## Example user template template
2  ## Example user template
3
4  # IntelliJ project files
5  .idea
6  *.iml
7  out
8  gen
9  ## Python template
10 # Byte-compiled / optimized / DLL files
11 __pycache__
12 *.py[cod]
13 *.py.class
14
15 # C extensions
16 *.so
17
18 # Distribution / packaging
19 .Python
20 build/
21 develop-eggs/
22 dist/
23 downloads/
24 eggs/
25 .eggs/
26 lib/
27 lib64/
28 parts/
29 sdist/
30 var/
31 wheels/
32 share/python-wheels/
33 *.egg-info/
34 .installed.cfg
35 *.egg
36 MANIFEST
37
38 # PyInstaller
39 # Usually these files are written by a python script from a template
40 # before PyInstaller builds the exe, so as to inject date/other infos into it.
41 *.manifest
42 *.spec
43
44 # Installer logs
45 win-inst.txt

```

Рисунок 4 – Изменения в gitignore

2. Была создана папка PyCharm в которой хранятся примеры из лабораторной работы.

| Имя        | Дата изменения   | Тип                    | Раз |
|------------|------------------|------------------------|-----|
| .git       | 02.12.2022 18:54 | Папка с файлами        |     |
| PyCharm    | 02.12.2022 19:01 | Папка с файлами        |     |
| .gitignore | 02.12.2022 18:41 | Исходный файл Git I... |     |
| LICENSE    | 02.12.2022 18:41 | Файл                   |     |

Рисунок 5 – Папка для хранения примеров

```
C:\Users\мвидео\PycharmProjects\pythonProject49\venv\Script
x = {'o', 'e', 'd', 'j', 'k'}
y = {'f', 'h', 'o', 'g', 'y', 'v', 'c'}

Process finished with exit code 0
```

Рисунок 6 – Результат работы программы

3. Были выполнены задания из лабораторной работы.

Задание 1.

Подсчитайте количество гласных в строке, введенной с клавиатуры с использованием множеств.

Код программы:

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

if __name__ == '__main__':
    s = input("Enter a sentence: ").lower()
    a = set(s.replace(' ', ''))
    vowels = set("aoueiy")
    count = 0
    for i in s:
        if i in vowels:
            count += 1
    print('The total number of vowels in the sentence is = ', count)
```

```
C:\Users\мвидео\PycharmProjects\pythonProject50\venv\Scripts\python
Enter a sentence: hello world
The total number of vowels in the sentence is = 3

Process finished with exit code 0
```

Рисунок 7 – Результат работы программы

## Задание 2.

Определите общие символы в двух строках, введенных с клавиатуры.

Код программы:

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

if __name__ == '__main__':
    s = input('enter a sentence: ')
    p1 = set(s.replace(' ', ''))
    s = input('enter a sentence: ')
    p2 = set(s.replace(' ', ''))
    print('The total amount of common symbols = ', p1.intersection(p2))
```

```
main x
C:\Users\мвидео\PycharmProjects\pythonProject51\venv\Scripts\python.exe C:\Users\мвидео\PycharmPro
enter a sentence: i can run faster with no wind resistance
enter a sentence: i do not know it ends i cannot see it in the distance
The total amount of common symbols = {'i', 's', 'd', 'a', 'e', 'n', 'o', 'h', 'w', 'c', 't'}

Process finished with exit code 0
```

Рисунок 8 – Результат работы программы

## Индивидуальное задание

### Вариант 5

$$5. \quad X = (A/B) \cap (C \cup D); \quad Y = (A \cap \bar{B}) \cup (C/D).$$

$$A = \{a, d, k, l, o, s\}; \quad B = \{d, e, k, s, u, x\}; \quad C = \{o, p, w\}; \quad D = \{d, n, r, y, z\};$$

Код программы:

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

if __name__ == '__main__':
    u = set("abcdefghijklmnopqrstuvwxyz")

    a = set('adklos')
```

```

b = set('deksux')
c = set('opw')
d = set('dnryz')

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

bn = u.difference(b)

y = (a.intersection(bn)).union(c.difference(d))
print(f"y = {y}")

```

```

C:\Users\мвидео\PycharmProjects\pythonProject52\venv\Scripts\pyth
x = {'o'}
y = {'w', 'o', 'a', 'l', 'p'}

Process finished with exit code 0

```

Рисунок 9 – Результат работы программы

```

c:\lbrt_2.7\lbrt_2.7>git add .
c:\lbrt_2.7\lbrt_2.7>git commit -m "fixation in doc"
[develop ea33c51] fixation in doc
6 files changed, 60 insertions(+)
create mode 100644 PyCharm/examp1.py
create mode 100644 "doc/\320\236\320\237\320\230_\320\273\320\260\320\26110.docx"
create mode 100644 "doc/\320\236\320\237\320\230_\320\273\320\260\320\26110.pdf"
create mode 100644 indiv/indiv1.py
create mode 100644 tasks/task2.py
create mode 100644 tasks/tasks.py
c:\lbrt_2.7\lbrt_2.7>git push --set-upstream origin develop
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 8 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (16/16), 2.22 MiB | 365.00 KiB/s, done.
Total 16 (delta 1), reused 3 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/meiokava/lbrt_2.7/pull/new/develop
remote:
To https://github.com/meiokava/lbrt_2.7.git
 * [new branch]      develop -> develop
branch 'develop' set up to track 'origin/develop'.

```

```

c:\lbrt_2.7\lbrt_2.7>git push --set-upstream origin develop
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 8 threads
Compressing objects: 100% (14/14), done.
Writing objects: 100% (16/16), 2.22 MiB | 365.00 KiB/s, done.
Total 16 (delta 1), reused 3 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/meiokava/lbrt_2.7/pull/new/develop
remote:
To https://github.com/meiokava/lbrt_2.7.git
 * [new branch]      develop -> develop
branch 'develop' set up to track 'origin/develop'.

```

Рисунок 10 – коммит и пуш изменений

```

c:\lbrt_2.7\lbrt_2.7>git merge develop
Updating 85b5ae0..ea33c51
Fast-forward
 PyCharm/examp1.py
 ...20\237\320\230_\320\273\320\260\320\26110.docx"
 ...320\237\320\230_\320\273\320\260\320\26110.pdf"
 indiv/indiv1.py
 tasks/task2.py
 tasks/tasks.py
 6 files changed, 60 insertions(+)
 create mode 100644 PyCharm/examp1.py
 create mode 100644 "doc/\320\236\320\237\320\230_\320\273\320\260\320\26110.docx"
 create mode 100644 "doc/\320\236\320\237\320\230_\320\273\320\260\320\26110.pdf"
 create mode 100644 indiv/indiv1.py
 create mode 100644 tasks/task2.py
 create mode 100644 tasks/tasks.py
c:\lbrt_2.7\lbrt_2.7>git push
To https://github.com/meiokava/lbrt_2.7.git

```

Рисунок 11 – Слияние веток main и develop, а также пуш изменений

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

### Контрольные вопросы

1. Что такое множества в языке Python?

Это неупорядоченная совокупность уникальных элементов.

2. Как осуществляется создание множества в Python?

С помощью фигурных скобок. Пример: `a = {a, b, c, d}`

3. Как проверить присутствие/отсутствие элемента в множестве?

`<элемент> in <множество>` или `<элемент> not in <множество>`

4. Как выполнить перебор элементов множества?

С помощью цикла `for`

5. Что такое set comprehension?

Это метод для создания множеств из других итерируемых объектов

6. Как выполнить добавление элемента во множество?

С помощью метода `add()`

7. Как выполнить удаление одного или всех элементов множества?

Удаление одного элемента производится с помощью метода `remove()`, а удаление при помощи метода `clear()`

8. Как выполнить основные операции над множествами:

объединение, пересечение, разность?

Объединение: `union()`

Пересечение: `intersection()`

Разность: `difference()`

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

С помощью методов `issubset()` и `issuperset()`

10. Каково назначение множеств `frozenset`?

Множество, созданное с помощью этого ключевого слова, нельзя изменять.

11. Как осуществляется преобразование множеств в строку, список, словарь?

С помощью методов `dict()` и `list()`