

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 «Основы программной инженерии»

Выполнил:
Юрьев Илья Евгеньевич
2 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка
и сопровождение программного
обеспечения», очная форма обучения

(подпись)

Руководитель практики:
Богданов С.С., ассистент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: 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.

Ход выполнения работы:

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

Рисунок 1 – Создание репозитория с заданными настройками

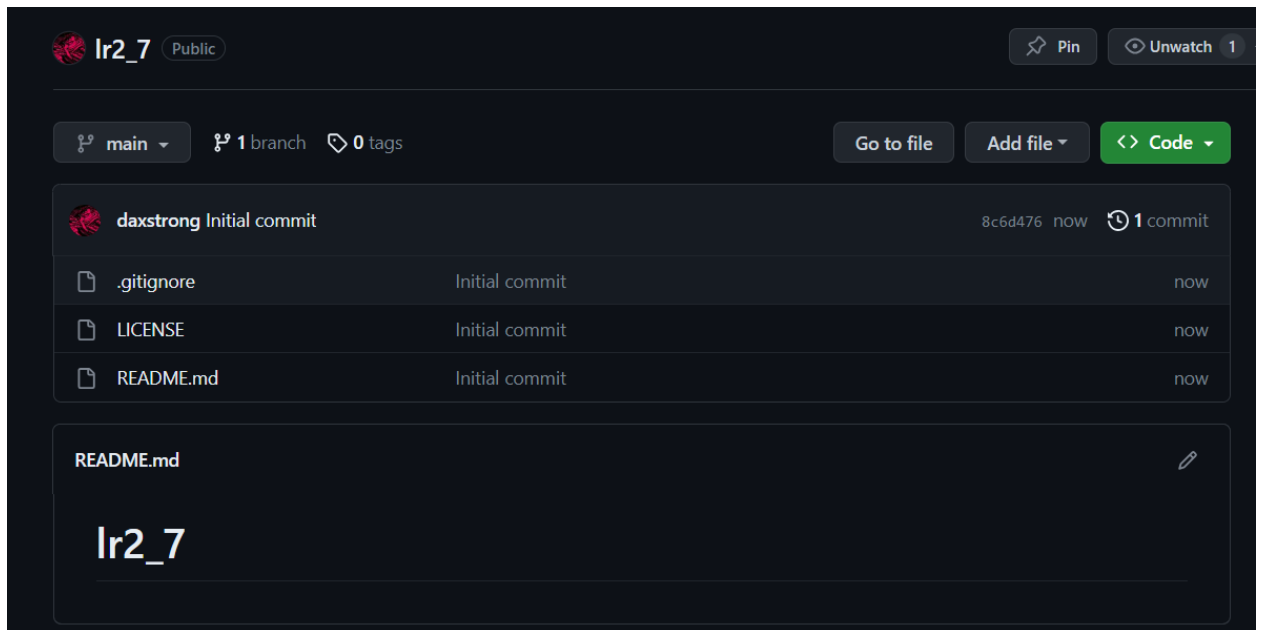


Рисунок 2 – Созданный репозиторий

```
ilyay@DESKTOP-FF1JT6S MINGW64 ~/OneDrive/Рабочий стол/Основы программной инженерии
$ git clone https://github.com/daxstrong/lr2_7.git
Cloning into 'lr2_7'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.
```

Рисунок 3 – Клонирование репозитория

```
ilyay@DESKTOP-FF1JT6S MINGW64 ~/OneDrive/Рабочий стол/Основы программной инженерии/lr2_7 (main)
$ git checkout -b develop
Switched to a new branch 'develop'
```

Рисунок 4 – Создание ветки develop

2. Проработать примеры лабораторной работы, оформляя код согласно ПЕР-8:

```

example1.py x
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4  ▶  if __name__ == "__main__":
5      # Определим универсальное множество
6      u = set("abcdefghijklmnopqrstuvwxyz")
7
8      a = {"b", "c", "h", "o"}
9      b = {"d", "f", "g", "o", "v", "y"}
10     c = {"d", "e", "j", "k"}
11     d = {"a", "b", "f", "g"}
12
13     x = (a.intersection(b)).union(c)
14     print(f"x = {x}")
15
16     # Найдем дополнения множеств
17     bn = u.difference(b)
18     cn = u.difference(c)
19
20     y = (a.difference(d)).union(cn.difference(bn))
21     print(f"y = {y}")
22

```

Рисунок 5 – Пример №1

```

example1 x
C:\Users\ilyay\AppData\Local\Microsoft\WindowsApps\python3.11.exe
x = {'k', 'd', 'o', 'j', 'e'}
y = {'g', 'f', 'c', 'o', 'v', 'h', 'y'}
Process finished with exit code 0

```

Рисунок 6 – Вывод программы (Пример №1)

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

```

1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  ▶  if __name__ == "__main__":
5      vowels = {'a', 'e', 'i', 'o', 'u', 'y'}
6      user_input = input("Введите строку: ").lower() # Получаем строку от пользователя и приводим к нижнему регистру
7      vowel_count = sum(1 for char in user_input if char in vowels) # Подсчитываем количество гласных
8      result = vowel_count
9
10     print(f"Количество гласных букв в строке: {result}")
11

```

Рисунок 7 – Количество гласных в строке (Задание №1)

```

task1 ×
C:\Users\ilyay\AppData\Local\Microsoft\WindowsApps\python3.11.exe "C:/Users/ilyay/OneDrive/Рабочий стол
Введите строку: tickets
Количество гласных букв в строке: 2
Process finished with exit code 0

```

Рисунок 8 – Вывод программы (Задание №1)

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

```

task2.py ×
1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  ▶  if __name__ == "__main__":
5      # Ввод двух строк с клавиатуры
6      string1 = input("Введите первую строку: ")
7      string2 = input("Введите вторую строку: ")
8
9      # Преобразование строк в множества уникальных символов
10     set1 = set(string1)
11     set2 = set(string2)
12
13     # Определение общих символов
14     common_characters = set1.intersection(set2)
15
16     print(f"Общие символы в двух введенных строках: {common_characters}")

```

Рисунок 9 – Общие символы (Задание №2)

```
task2 x
C:\Users\ilyay\AppData\Local\Microsoft\WindowsApps\python3.11.exe "C:/Users/ilyay/
Введите первую строку: abob
Введите вторую строку: borya
Общие символы в двух введенных строках: {'o', 'a', 'b'}

Process finished with exit code 0
```

Рисунок 10 – Вывод программы (Задание №2)

5. Выполним индивидуальное задание:

12.
$$A = \{b, k, n, o, q\}; \quad B = \{a, b, k, u\}; \quad C = \{o, p\}; \quad D = \{a, m, n, y, z\};$$
$$X = (A \cup B) \cap D; Y = (\bar{A} \cap D) \cup (C/B).$$

```
individual1.py x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  if __name__ == "__main__":
5      a = {'b', 'k', 'n', 'o', 'q'}
6      b = {'a', 'b', 'k', 'u'}
7      c = {'o', 'p'}
8      d = {'a', 'm', 'n', 'y', 'z'}
9
10     x = (a.union(b)).intersection(d)
11     y = (set()).intersection(d).union(c.difference(b))
12
13     if not x:
14         print("x - пустое множество")
15     else:
16         print(f'x = {x}')
17
18     if not y:
19         print("y - пустое множество")
20     else:
21         print(f'y = {y}')
```

Рисунок 11 – Решение индивидуального задания

```
individual1 x
C:\Users\ilyay\AppData\Local\Microsoft\WindowsApps\python3.11.exe
x = {'n', 'a'}
y = {'o', 'p'}
```

Рисунок 12 – Вывод программы (Индивидуальное задание)

6. Зафиксируем сделанные изменения, сольем ветки и отправим на удаленный репозиторий:

```
ilyay@DESKTOP-FF1JT6S MINGW64 ~/OneDrive/Рабочий стол/Основы программной инженерии/lr2_7 (develop)
$ git log
commit 38c0568fb66132bf3ec23f3975281f9b572da732 (HEAD -> develop)
Author: dexstrong <ilya.yurev.04@inbox.ru>
Date: Tue Dec 5 02:11:38 2023 +0300

    final changes

commit 8c6d476e91884bd0817de2f18336b459c65d474f (origin/main, origin/HEAD, main)
Author: Ilya Yurev <112946692+daxstrong@users.noreply.github.com>
Date: Tue Dec 5 01:21:56 2023 +0300

    Initial commit
```

Рисунок 13 – Коммиты ветки develop во время выполнения лабораторной Работы

```
ilyay@DESKTOP-FF1JT6S MINGW64 ~/OneDrive/Рабочий стол/Основы программной инженерии/lr2_7 (develop)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

ilyay@DESKTOP-FF1JT6S MINGW64 ~/OneDrive/Рабочий стол/Основы программной инженерии/lr2_7 (main)
$ git merge develop
Updating 8c6d476..38c0568
Fast-forward
 .idea/.gitignore | 8 ++++++
 .idea/inspectionProfiles/profiles_settings.xml | 6 +++++
 .idea/lr2_7.iml | 8 ++++++
 .idea/misc.xml | 4 ++++
 .idea/modules.xml | 8 ++++++
 .idea/vcs.xml | 6 ++++++
 example1.py | 21 ++++++++++++++++++++++++++++++++++++++
 individual1.py | 21 ++++++++++++++++++++++++++++++++++++++
 task1.py | 10 ++++++++
 task2.py | 17 ++++++++++++++++++++++++++++++++++++++
10 files changed, 109 insertions(+)
create mode 100644 .idea/.gitignore
create mode 100644 .idea/inspectionProfiles/profiles_settings.xml
create mode 100644 .idea/lr2_7.iml
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 example1.py
create mode 100644 individual1.py
create mode 100644 task1.py
create mode 100644 task2.py
```

Рисунок 14 – Слияние веток main и develop

```

ilyay@DESKTOP-FF1JT6S MINGW64 ~/OneDrive/Рабочий стол/Основы программной инженерии/lr2_7 (main)
$ git push origin main
Enumerating objects: 15, done.
Counting objects: 100% (15/15), done.
Delta compression using up to 12 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (14/14), 2.91 KiB | 2.91 MiB/s, done.
Total 14 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/daxstrong/lr2_7.git
 8c6d476..38c0568  main -> main

```

Рисунок 15 – Отправка изменений на удаленный репозиторий

The screenshot shows the GitHub interface for a repository named 'lr2_7' (Public). The main branch is 'main'. A commit by 'daxstrong' with the message 'final changes' is displayed, dated '2 minutes ago' with commit hash '38c0568' and '2 commits'. Below the commit message is a table of files:

File	Commit Message	Time
.idea	final changes	2 minutes ago
.gitignore	Initial commit	51 minutes ago
LICENSE	Initial commit	51 minutes ago
README.md	Initial commit	51 minutes ago
example1.py	final changes	2 minutes ago
individual1.py	final changes	2 minutes ago
task1.py	final changes	2 minutes ago
task2.py	final changes	2 minutes ago

Below the file list, the 'README.md' content is shown, which includes the text 'lr2_7'.

Рисунок 16 – Изменения удаленного репозитория

Ответы на контрольные вопросы:

1. Что такое множества в языке Python?

Множество (set) в Python – это неупорядоченная коллекция уникальных элементов. Они используются для выполнения операций над уникальными элементами без дублирования.

2. Как осуществляется создание множеств в Python?

Множество можно создать, используя фигурные скобки {} и перечислив элементы множества через запятую. Например:

```
my_set = {1, 2, 3, 4}
```

3. Как проверить присутствие/отсутствие элемента в множестве?

Для проверки присутствия элемента в множестве можно использовать оператор in. Например:

```
my_set = {1, 2, 3, 4}
```

```
print(3 in my_set) # Выведет: True
```

```
print(5 not in my_set) # Выведет: True
```

4. Как выполнить перебор элементов множества?

Можно использовать цикл for для перебора элементов множества:

```
my_set = {1, 2, 3, 4}
```

```
for element in my_set:
```

```
    print(element)
```

5. Что такое set comprehension?

Set comprehension – это способ создания множества с использованием компактного синтаксиса, аналогичного списочным включениям. Например:

```
my_set = {x for x in range(10) if x % 2 == 0} # Создание множества четных чисел от 0 до 9
```

6. Как выполнить добавление элемента во множество?

Для добавления элемента в множество используется метод .add().

Например:

```
my_set = {1, 2, 3}
```

```
my_set.add(4)
```

7. Как выполнить удаление одного или всех элементов множества?

Методы `.remove()` и `.discard()` используются для удаления одного элемента, а метод `.clear()` - для удаления всех элементов множества. Например:

```
my_set = {1, 2, 3, 4}
```

```
my_set.remove(3)
```

```
my_set.discard(5) # Если элемент отсутствует, discard() не вызывает ошибку
```

```
my_set.clear() # Очистка множества
```

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

Объединение: `set1.union(set2)` или оператор `|`

Пересечение: `set1.intersection(set2)` или оператор `&`

Разность: `set1.difference(set2)` или оператор `-`

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

Методы `.issubset()` и `.issuperset()` используются для определения того, чтобы определить является ли одно множество подмножеством или надмножеством другого соответственно.

10. Каково назначение множеств `frozenset`?

`frozenset` – это неизменяемая версия множества. Однажды созданное `frozenset` не может быть изменено, но оно может быть использовано в качестве ключа в словарях или как элемент в другом множестве.

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

Для преобразования множества в список можно использовать `list(my_set)`. Преобразование в строку: `str(my_set)`.

Множество не может быть преобразовано непосредственно в словарь, но можно создать словарь из множества ключей.