

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 №10
дисциплины «Основы программной инженерии»

Выполнил:
Магомедов Имран Борисович
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:
Воронкин Р.А., кандидат технических
наук, доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____

Дата защиты _____

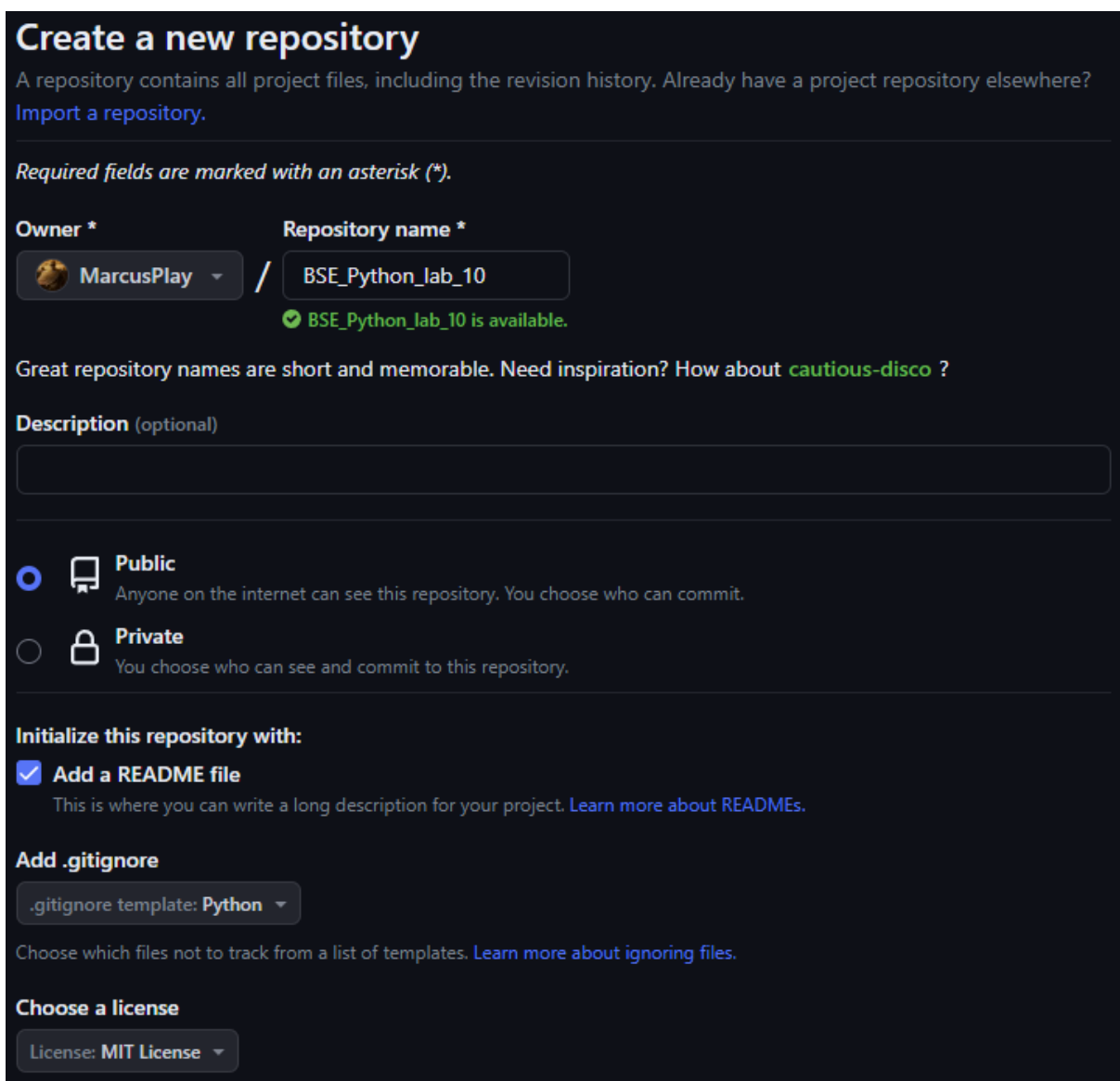
Ставрополь, 2023 г.

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.

Методика и порядок выполнения работы

1. Изучить теоретический материал работы.
2. 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 explains that a repository contains all project files, including the revision history. Below this, it asks if the user already has a project repository elsewhere and provides a link to 'Import a repository'. A note states 'Required fields are marked with an asterisk (*)'. The form has two main sections: 'Owner' and 'Repository name'. The 'Owner' is set to 'MarcusPlay' with a dropdown arrow. The 'Repository name' is 'BSE_Python_lab_10' with a dropdown arrow. A green checkmark indicates 'BSE_Python_lab_10 is available'. Below this, there is a suggestion for repository names: 'Great repository names are short and memorable. Need inspiration? How about cautious-disco?'. There is a 'Description (optional)' text area. The 'Visibility' section has two options: 'Public' (selected with a radio button) and 'Private'. The 'Initialize this repository with:' section has a checked checkbox for 'Add a README file'. Below this is a link to 'Learn more about READMEs'. The 'Add .gitignore' section has a dropdown menu set to 'Python'. Below this is a link to 'Learn more about ignoring files'. The 'Choose a license' section has a dropdown menu set to 'MIT License'.

3. Выполните клонирование созданного репозитория.

```
Admin@Genuine MINGW64 ~/Desktop/Work
$ git clone https://github.com/MarcusPlay/BSE_Python_lab_10.git
Cloning into 'BSE_Python_lab_10'...
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.
```

4. Дополните файл .gitignore необходимыми правилами.
5. Организуйте свой репозиторий в соответствие с моделью ветвления git-flow.

```
Admin@Genuine MINGW64 ~/Desktop/Work/BSE_Python_lab_10 (main)
$ git checkout -b develop
Switched to a new branch 'develop'
```

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

```
PS C:\Users\Admin\Desktop\Work\BSE_Python_lab_10> git commit -m "added folder examples"
[develop 2e078a4] added folder examples
1 file changed, 21 insertions(+)
create mode 100644 examples/example_1.py
```

Пример 1.

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  v 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}")

```

```

x = {'o', 'j', 'e', 'd', 'k'}
y = {'c', 'f', 'v', 'g', 'o', 'y', 'h'}

```

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

```

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

# подсчитайте количество гласных в строке, введенной с клавиатуры с использованием множеств.

if __name__ == "__main__":
    input_string = input("Введите строку: ").lower()

    set_value = set("aeiouaeёиюуэюя")

    count = sum(1 for i in input_string if i in set_value)

    print("Количество гласных в введенной строке:", count)

```

8. Зафиксируйте сделанные изменения в репозитории.

```
PS C:\Users\Admin\Desktop\Work\BSE_Python_lab_10> git add .
PS C:\Users\Admin\Desktop\Work\BSE_Python_lab_10> git commit -m "added task_1.py"
[develop 262a081] added task_1.py
1 file changed, 15 insertions(+)
create mode 100644 task_1.py
```

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

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

# подсчитайте количество гласных в строке, введенной с клавиатуры с использованием множеств.

if __name__ == "__main__":
    string_1 = set(input("Введите строку 1: "))
    string_2 = set(input("Введите строку 2: "))

    identical_characters = string_1.intersection(string_2)

    print(identical_characters)
```

10. Зафиксируйте сделанные изменения в репозитории.

```
PS C:\Users\Admin\Desktop\Work\BSE_Python_lab_10> git add .
PS C:\Users\Admin\Desktop\Work\BSE_Python_lab_10> git commit -m "added task_2.py"
[develop 84ee5b7] added task_2.py
1 file changed, 16 insertions(+)
create mode 100644 task_2.py
```

11. Сделать индивидуальное задание.

Вариант 12. Определить результат выполнения операций над множествами. Считать элементы множества строками. Проверить результаты вручную.

$$X = (A \cup B) \cap D; Y = (\neg A \cap D) \cup (C/B)$$

$$A = \{a, b, g, k, m, p\}; B = \{b, e, f, l, r\}; C = \{k, l, w, x\}; D = \{e, j, o, p, q, u, v\}$$

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

#Определить результат выполнения операций над множествами.
#Считать элементы множества строками. Проверить результаты вручную.
#  $X=(A \cup B) \cap D$ ;  $Y=(\neg A \cap D) \cup (C/B)$ 
#  $A=\{a,b,g,k,m,p\}$ ;  $B=\{b,e,f,l,r\}$ ;  $C=\{k,l,w,x\}$ ;  $D=\{e,j,o,p,q,u,v\}$ 

if __name__ == "__main__":
    A = set('abgkmp')
    B = set('beflr')
    C = set('klwx')
    D = set('ejopquv')
    U = set('abgkmpbeflrklwxejopquv')

    X = (A.union(B)).intersection(D)
    Y = ((U.difference(A)).intersection(D)).union(C.difference(B))

    print(X, Y)
```

```
{'e', 'p'} {'v', 'e', 'j', 'q', 'x', 'w', 'u', 'o', 'k'}
```

12. Добавьте отчет по лабораторной работе в формате PDF в папку doc репозитория. Зафиксируйте изменения.
13. Выполните слияние ветки для разработки с веткой master/main.
14. Отправьте сделанные изменения на сервер GitHub.
15. Отправьте адрес репозитория GitHub на электронный адрес преподавателя.

Контрольные вопросы.

1. Что такое множества в языке Python? Множество - это неупорядоченная коллекция уникальных элементов.
2. Как осуществляется создание множеств в Python? Множество создаются с использованием фигурных скобок {}: `my_set = {1, 2, 3}` или функции `set()`: `my_set = set([1, 2, 3])`.

3. Как проверить присутствие/отсутствие элемента в множестве?

В python можно проверить присутствие элемента в множестве с использованием оператора `in`. Например:

```
my_set = {1, 2, 3, 4}
if 4 in my_set:
    print('Элемент 4 присутствует в множестве.')
```

4. Как выполнить перебор элементов множества? Для перебора элементов в множестве в Python вы можете использовать цикл `for`. Например:

```
my_set = {1, 2, 3, 4, 5}
for element in my_set:
    print(element)
```

5. Что такое `set comprehension`? `Set comprehension` - это синтаксическая конструкция, позволяющая создавать множества с использованием более компактного кода. Например:

```
my_set = {x for x in range(1, 6)}
#Результат: {1, 2, 3, 4, 5}
```

6. Как выполнить добавление элемента во множество? Чтобы добавить элемент в множество в Python, используйте метод `add()`. Например:

```
my_set = {1, 2, 3}
my_set.add(4)
#Результат: {1, 2, 3, 4}
```

7. Как выполнить удаление одного или всех элементов множества? Для удаления одного элемента из множества используется `remove()`. Если вы хотите удалить все элементы из множеств, используйте метод `clear()`. Например:

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

```
my_set.remove(3)
#Результат: {1, 2, 4}
my_set.clear()
#Результат: {}
```

8. Как выполняются основные операции над множествами: объединение, пересечение, разность? В Python для выполнения основных операций над множествами используются следующие операторы и методы:

→ Объединение:

- ◆ Оператор |
- ◆ Метод `union()`

Например:

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
result_union = set1 | set2
result_union_method = set1.union(set2)
#Результат: {1, 2, 3, 4, 5}
```

→ Пересечение:

- ◆ Оператор &
- ◆ Метод `intersection()`

Например:

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
result_intersection = set1 & set2
result_intersection_method =
set1.intersection(set2)
#Результат: {3}
```

→ Разность

- ◆ Оператор -
- ◆ Метод `difference()`

Например:

```
set1 = {1, 2, 3}
set2 = {3, 4, 5}
result_difference = set1 - set2
result_difference_method = set1.difference(set2)
#Результат: {1, 2}
```

9. Как определить, что некоторое множество является надмножеством или подмножеством другого множества?
10. Каково назначение множеств `frozenset` ?
11. Как осуществляется преобразование множеств в строку, список, словарь.