

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 №7
дисциплины «Основы программной инженерии»

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

(подпись)

Доцент кафедры инфокоммуникаций
Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

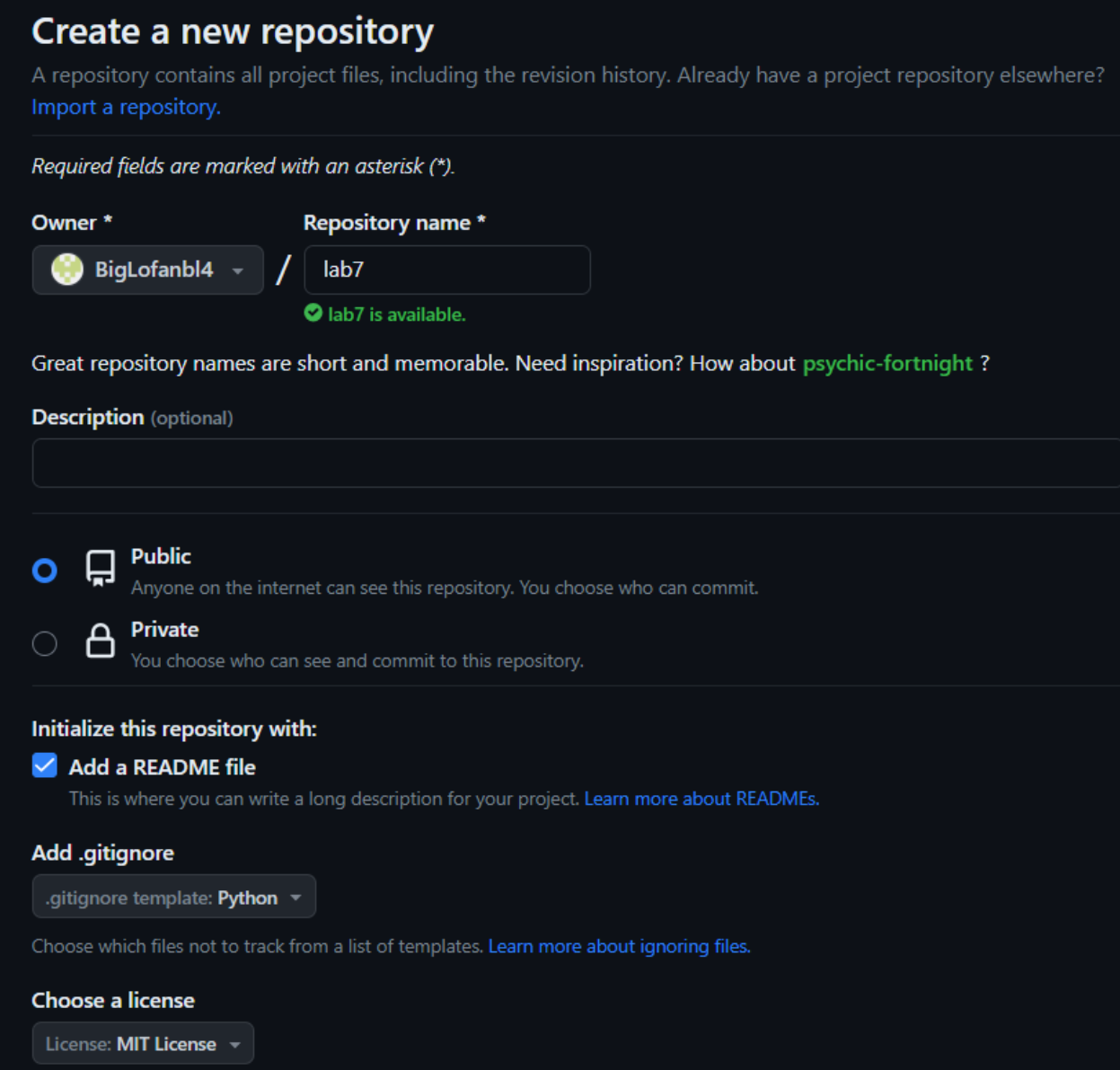
Ставрополь, 2023 г.

Тема: работа со списками в языке Python.

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

Порядок выполнения работы

1. Создал репозиторий GitHub.




Create a new repository

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

Required fields are marked with an asterisk ().*

Owner * **Repository name ***

 BigLofanbl4 / lab7

✔ lab7 is available.

Great repository names are short and memorable. Need inspiration? How about **psychic-fortnight** ?

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:

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

Add .gitignore

.gitignore template: **Python**

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: **MIT License**

Рисунок 1 – Создание репозитория GitHub

2. Проработал примеры из лабораторной работы.

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

import sys

if __name__ == '__main__':

    # Ввести список одной строкой.
    A = list(map(int, input().split()))

    # Проверить количество элементов списка.
    if len(A) != 10:
        print("Неверный размер списка", file=sys.stderr)
        exit(1)

    # Найти искомую сумму.
    s = 0
    for item in A:
        if abs(item) < 5:
            s += item

    print(s)
```

Рисунок 2.1 – Код из примера 1

```
2 3 4 1 5 6 -1 0 4 5
13
```

Рисунок 2.2 – Вывод программы из примера 1

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

import sys

if __name__ == '__main__':
    # Ввести список одной строкой.
    a = list(map(int, input().split()))

    # Если список пуст, завершить программу.
    if not a:
        print("Заданный список пуст", file=sys.stderr)
        exit(1)

    # Определить индексы минимального и максимального элементов.
    a_min = a_max = a[0]
    i_min = i_max = 0
    for i, item in enumerate(a):
        if item < a_min:
            i_min, a_min = i, item
        if item >= a_max:
            i_max, a_max = i, item

    # Проверить индексы и обменять их местами.
    if i_min > i_max:
        i_min, i_max = i_max, i_min

    # Посчитать количество положительных элементов.
    count = 0
    for item in a[i_min+1:i_max]:
        if item > 0:
            count += 1

    print(count)
```

Рисунок 2.3 – Код из примера 2

```
3 2 4 1 5 6 1 3 -3 -4 -4
2
```

Рисунок 2.4 – Вывод программы из примера 2

3. Ввести список А из 10 элементов, найти разность положительных элементов кратных 11, их количество и вывести результаты на экран.

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

if __name__ == "__main__":
    a = [int(input("Enter number: ")) for i in range(10)]
    b = []

    for num in a:
        if num % 11 == 0 and num > 0:
            b.append(num)

    diff = b[0]
    for num in b[1:]:
        diff -= num

    print(f"Differnce {diff}")
    print(f"Count: {len(b)}")
```

Рисунок 3.1 – Код программы

```
Enter number: 11
Enter number: 11
Enter number: 22
Enter number: 1
Enter number: -11
Enter number: -22
Enter number: 3
Enter number: 2
Enter number: 5
Enter number: 3
Differnce -22
Count: 3
```

Рисунок 3.2 – Вывод программы

4. В списке, состоящем из вещественных элементов, вычислить:
 - количество элементов списка, больших C;
 - произведение элементов списка, расположенных после максимального по модулю элемента.

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

if __name__ == "__main__":
    lst = [float(input("Enter elemnt of list: ")) for i in range(10)]
    c = float(input("Enter C: "))

    # 1 задача: количество элементов списка, больших C;
    count = 0
    for elem in lst:
        if elem > c:
            count += 1
    print(f"Count of elements greater than {c}: {count}")

    # 2 задача: произведение элементов списка, расположенных после максимального по модулю элемента.
    maxElemIndex = lst.index(max(lst))
    product = 1
    for elem in lst[maxElemIndex + 1 :]:
        product *= elem
    print(f"Product: {product}")

    # Преобразование списка
    lst.sort(key=lambda x: x >= 0)
    print(lst)
```

Рисунок 4.1 – Код программы

```
Enter elemnt of list: -8
Enter elemnt of list: -17.5
Enter elemnt of list: 23
Enter elemnt of list: 19
Enter elemnt of list: 23.4
Enter elemnt of list: 0
Enter elemnt of list: 0
Enter elemnt of list: -4.4
Enter elemnt of list: 26.4
Enter elemnt of list: 2
Enter C: 10
Count of elements greater than 10.0: 4
Product: 2.0
[-8.0, -17.5, -4.4, 23.0, 19.0, 23.4, 0.0, 0.0, 26.4, 2.0]
```

Рисунок 4.2 – Вывод программы

Ответы на контрольные вопросы

1. Список – это структура данных, которая хранит различные объекты.
2. Создание списка осуществляется с помощью квадратных скобок:
$$a = []$$
3. Переменная, которая объявлена как список, хранит ссылку на структуру (можно сказать контейнер) в памяти, которая в свою очередь хранит ссылки на элементы данных.
4. Перебрать все элементы списка можно с помощью циклов.
5. Списки можно складывать (конкатенация) и умножать на число (в данном случае список повторится n раз).
6. С помощью оператора `in`.
7. С помощью метода `count`.
8. Вставка – метод `insert()`, который принимает в качестве аргументов индекс, по которому вставляется, и сам вставляемый элемент; добавление элемента – метод `append`.
9. С помощью метода `sort()`.
10. Удалить один элемент: методы `pop`, `remove`. Удалить несколько при помощи функции `del` и среза.
11. Списковое включение – это способ построения списков `[i for i in range(10)]`, можно также использовать условие, для обработки.
12. `[start:stop:step]`
13. `min()`, `max()`, `sum()`, `len()`.
14. С помощью функции `sorted()` или среза `[:]`.
15. Функция `sorted` может применяться не только к спискам, и она возвращает новый список.