

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №7
дисциплины
«Основы кроссплатформенного программирования»

Выполнил:
Ляш Денис Александрович
2 курс, группа ИТС-б-о-23-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи», очная
форма обучения

(подпись)

Проверил:
Доцент департамента цифровых,
робототехнических систем и
электроники Воронкин Р.А.

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2024 г.

Тема: Работа со списками в языке Python

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

Порядок выполнения работы:

1. Изучил теоретический материал.
2. Приступил к выполнению заданий.
3. Создание репозитория

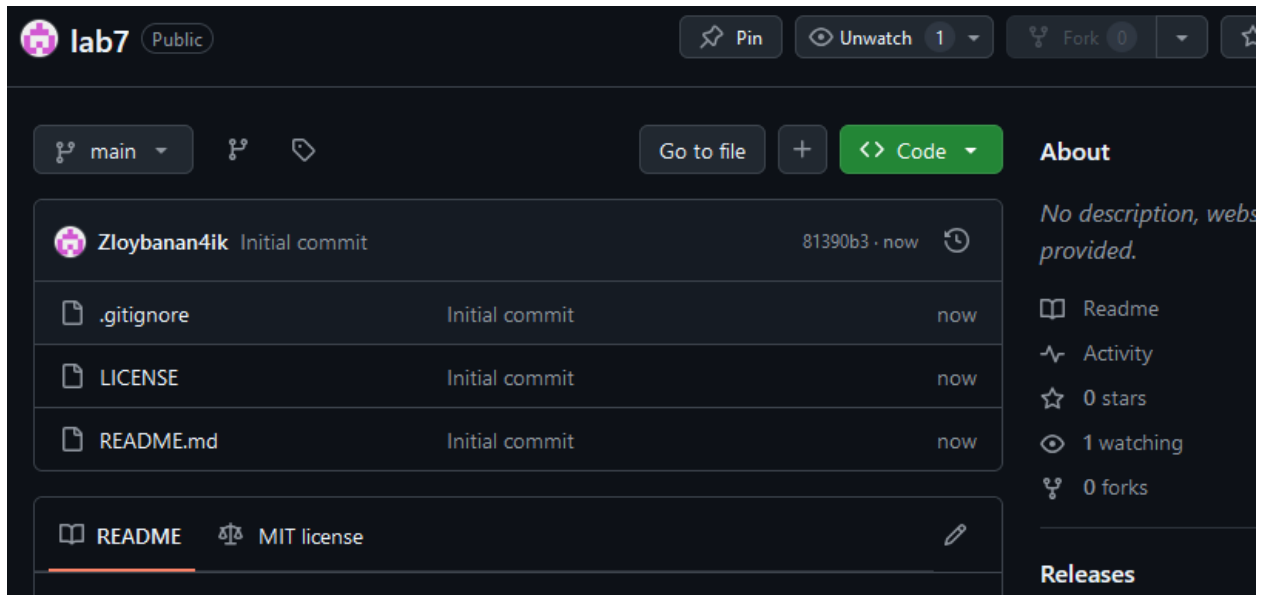


Рисунок 1. Репозиторий

4. Клонирование репозитория

```
user@DESKTOP-VI4I94J MINGW64 ~/workspace
$ git clone https://github.com/Zloybanan4ik/lab7.git
Cloning into 'lab7'...
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 (from 0)
Receiving objects: 100% (5/5), done.

user@DESKTOP-VI4I94J MINGW64 ~/workspace
$ cd lab7
```

Рисунок 2. Клонирование

5. Проработал примеры

```
#!/usr/bin/env python3
```

```

# -*- coding: utf-8 -*-

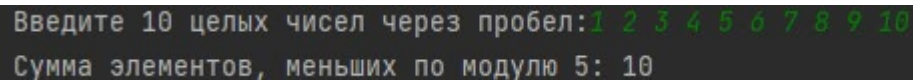
if __name__ == '__main__':
    # Ввод списка из 10 элементов
    A = list(map(int, input("Введите 10 целых чисел через
    пробел:").split()))

    # Проверка, что введено именно 10 элементов
    if len(A) != 10:
        print("Ошибка: необходимо ввести ровно 10 целых чисел.",
        file=sys.stderr)
        exit(1)

    # Вычисление суммы элементов, меньших по модулю 5
    total_sum = sum(item for item in A if abs(item) < 5)

    # Вывод суммы
    print("Сумма элементов, меньших по модулю 5:", total_sum)

```



```

Введите 10 целых чисел через пробел:1 2 3 4 5 6 7 8 9 10
Сумма элементов, меньших по модулю 5: 10

```

Рисунок 3. Пример 1

```

def count_positive_between_min_max(lst):
    if not lst:
        return 0 # Если список пустой, возвращаем 0

    min_value = min(lst)
    max_value = max(lst)

    # Находим индексы минимального и максимального элементов
    min_index = lst.index(min_value)
    max_index = lst.index(max_value)

    # Убедимся, что индексы корректные (первый индекс - меньший)
    start_index = min(min_index, max_index) + 1
    end_index = max(min_index, max_index)

    # Считаем количество положительных элементов между ними
    positive_count = sum(1 for x in lst[start_index:end_index] if x > 0)

    return positive_count

```

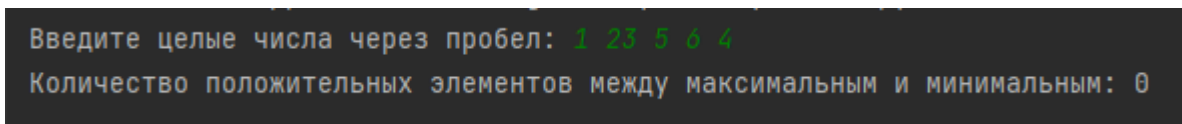
```

if __name__ == '__main__':
    # Ввод списка целых чисел
    A = list(map(int, input("Введите целые числа через пробел: ").split()))

    # Получаем количество положительных элементов между минимальным и
    максимальным
    result = count_positive_between_min_max(A)

    print("Количество положительных элементов между максимальным и
    минимальным:", result)

```



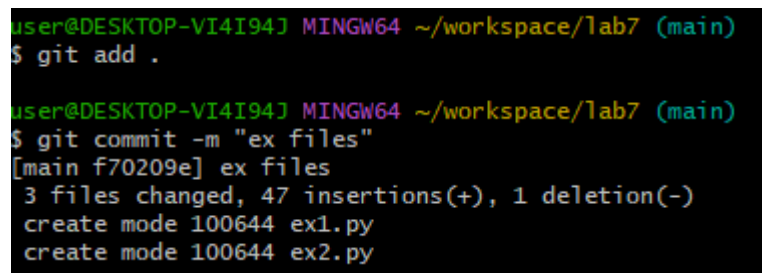
```

Введите целые числа через пробел: 1 23 5 6 4
Количество положительных элементов между максимальным и минимальным: 0

```

Рисунок 4. Пример 2

6. Зафиксировал изменения



```

user@DESKTOP-VI4I94J MINGW64 ~/workspace/lab7 (main)
$ git add .

user@DESKTOP-VI4I94J MINGW64 ~/workspace/lab7 (main)
$ git commit -m "ex files"
[main f70209e] ex files
3 files changed, 47 insertions(+), 1 deletion(-)
create mode 100644 ex1.py
create mode 100644 ex2.py

```

Рисунок 5. Изменения

7. Приступил к выполнению индивидуального задания

```

# list_sum_count.py

# Ввод списка A из 10 элементов
A = [int(input(f"Enter element {i+1} of list A: ")) for i in range(10)]

# Фильтрация элементов, удовлетворяющих условиям
filtered_elements = [x for x in A if 2 < x < 20 and x % 8 == 0]

# Подсчёт суммы и количества элементов
sum_filtered = sum(filtered_elements)
count_filtered = len(filtered_elements)

# Вывод результатов
print(f"Sum of elements greater than 2, less than 20, and divisible by 8:

```

```
{sum_filtered}")  
print(f"Count of such elements: {count_filtered}")
```

```
Enter element 1 of list A: 1  
Enter element 2 of list A: 2  
Enter element 3 of list A: 3  
Enter element 4 of list A: 4  
Enter element 5 of list A: 5  
Enter element 6 of list A: 6  
Enter element 7 of list A: 7  
Enter element 8 of list A: 9  
Enter element 9 of list A: 7  
Enter element 10 of list A: 4  
Sum of elements greater than 2, less than 20, and divisible by 8: 0  
Count of such elements: 0
```

Рисунок 6. Задание 1

```
# list_reordering.py
```

```
# Ввод списка вещественных чисел
```

```
n = int(input("Enter the number of elements in the list: "))
```

```
lst = [float(input(f"Enter element {i+1}: ")) for i in range(n)]
```

```
# Ввод диапазона A и B
```

```
A = float(input("Enter the lower bound (A): "))
```

```
B = float(input("Enter the upper bound (B): "))
```

```
# 1. Количество элементов в диапазоне от A до B
```

```
count_in_range = sum(A <= x <= B for x in lst)
```

```
# 2. Сумма элементов после максимального элемента
```

```
max_element = max(lst)
```

```
max_index = lst.index(max_element)
```

```
sum_after_max = sum(lst[max_index + 1:])
```

```
# 3. Упорядочивание элементов по убыванию модулей
```

```
lst_sorted = sorted(lst, key=abs, reverse=True)
```

```
# Вывод результатов
```

```
print(f"Count of elements in the range [{A}, {B}]: {count_in_range}")
```

```
print(f"Sum of elements after the maximum element ({max_element}):
```

```
{sum_after_max}")  
print(f"List sorted by descending absolute values: {lst_sorted}")
```

```
Enter the number of elements in the list: 5  
Enter element 1: 1  
Enter element 2: 2  
Enter element 3: 3  
Enter element 4: 4  
Enter element 5: 5  
Enter the lower bound (A): 0  
Enter the upper bound (B): 6  
Count of elements in the range [0.0, 6.0]: 5  
Sum of elements after the maximum element (5.0): 0  
List sorted by descending absolute values: [5.0, 4.0, 3.0, 2.0, 1.0]
```

Рисунок 7. Задание 2

8. Зафиксировал изменения

Ответы на контрольные вопросы:

1. Что такое списки в языке Python?

Списки в Python — это изменяемые последовательности, которые могут содержать элементы различных типов, включая числа, строки и другие объекты. Списки позволяют хранить и управлять коллекциями данных.

2. Как осуществляется создание списка в Python?

Список создается с помощью квадратных скобок [], также можно использовать функцию list.

3. Как организовано хранение списков в оперативной памяти?

Списки хранятся в виде массивов указателей на объекты. Каждый элемент списка хранит ссылку на объект, а не сам объект. Это позволяет спискам содержать элементы разных типов.

4. Каким образом можно перебрать все элементы списка?

Перебор элементов списка можно осуществить с помощью цикла for.

5. Какие существуют арифметические операции со списками?

Основные арифметические операции со списками: конкатенация— объединяет два списка. Повторение — повторяет список n раз.

6. Как проверить есть ли элемент в списке?

Через оператора `in` `if element in my list`

7. Как определить число вхождений заданного элемента в списке?

Через метод `count`

8. Как осуществляется добавление (вставка) элемента в список?

Через метод `append` для добавления в конец списка или `insert` для вставки по индексу.

9. Как выполнить сортировку списка?

Через `sort` для сортировки списка на месте или функцию `sorted` для создания нового отсортированного списка

10. Как удалить один или несколько элементов из списка?

Через `remove` для удаления первого вхождения элемента, `pop(index)` для удаления элемента по индексу, или `del` для удаления по индексу

11. Что такое списковое включение и как с его помощью осуществлять обработку списков?

Списковое включение — это способ создания нового списка на основе существующего.

12. Как осуществляется доступ к элементам списков с помощью срезов?

Срезы позволяют извлекать подсписки. Синтаксис: `my list[start:end]`, где `start` — начальный индекс, а `end` — конечный индекс.

13. Какие существуют функции агрегации для работы со списками?

Основные функции: `sum` — сумма элементов. `min` — минимальный элемент. `max` — максимальный элемент. `len` — количество элементов.

14. Как создать копию списка?

Через метод `copy`

15. Самостоятельно изучите функцию `sorted` языка Python. В чем ее отличие от метода `sort` списков?

Функция `sorted` создает новый отсортированный список и не изменяет оригинальный, а метод `sort` сортирует список на месте и возвращает `None`

Вывод: в ходе работы исследовал базовые возможности системы контроля версий Git для работы с локальными репозиториями.

Ссылка на репозиторий: <https://github.com/Zloybanan4ik/lab7.git>