

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 №2.4
дисциплины «Программирование на Python»

Выполнил:
Дзуев Альберт Мухаметович
2 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и
вычислительная техника»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем», очная
форма обучения

(подпись)

Руководитель практики:
Воронкин Р.А., доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

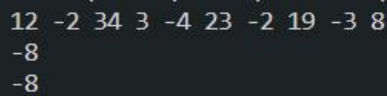
Ставрополь, 2023 г.

Тема: Работа со списками в языке Python

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

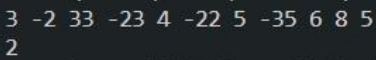
Порядок выполнения работы:

1. Создал новый репозиторий, клонировал его, в нем создал ветку developer и перешел на нее.
2. Проработал примеры лабораторной работы:



```
12 -2 34 3 -4 23 -2 19 -3 8
-8
-8
```

Рисунок 1. Вывод программы примера 1



```
3 -2 33 -23 4 -22 5 -35 6 8 5
2
```

Рисунок 2. Вывод программы примера 2

3. Выполнил индивидуальное задание 1 вариант 9: Составить программу, выдающую индексы заданного элемента или сообщаящую, что такого элемента в списке нет.

Код программы:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*- import sys
if __name__ == '__main__':
    print("Введите элементы списка через пробел: ")
    lst = list(map(int, input().split()))
    # Если список пуст, завершить программу.
    if not lst:
        print("Заданный список пуст", file=sys.stderr)
        exit(1)
    target = int(input("Введите элемент для поиска его индекса: "))
    if target in lst:
        index = [i for i, x in enumerate(lst) if x == target]
        print(f"Индекс элемента {target}: {index}")
    else:
        print(f"Элемент {target} не найден в
списке")
```

```

Введите элементы списка через пробел:
1 4 2 6 7
Введите элемент для поиска его индекса: 2
Индекс элемента 2: [2]
PS C:\Users\dzuev> & C:/Users/dzuev/AppData/Local/Microsoft/WindowsApps/python3.12.exe c:/Users/dzuev/Untitled-1.py
Введите элементы списка через пробел:
12 5 3 -4 6 -3
Введите элемент для поиска его индекса: 2
Элемент 2 не найден в списке

```

Рисунок 4. Несколько запусков программы индивидуального задания 1

4. Выполнил индивидуальное задание 2 вариант 9: В списке, состоящем из целых элементов, вычислить: 1. минимальный по модулю элемент списка; 2. сумму модулей элементов списка, расположенных после первого элемента, равного нулю. Преобразовать список таким образом, чтобы в первой его половине располагались элементы, стоявшие в четных позициях, а во второй половине - элементы, стоявшие в нечетных позициях.

Код программы:

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*- import math
import math
import sys
if __name__ == '__main__':
    print("Введите список целых чисел через пробел")
    input_list = list(map(int, input().split()))
    m_value = math.fabs(input_list[0])
    for i, num in enumerate(input_list):
        abs_value = math.fabs(num)
        if abs_value < m_value:
            m_value = abs_value
    summ = 0
    zero_found = False
    for num in input_list:
        if zero_found:
            summ += abs(num)
        elif num == 0:
            zero_found = True
    def rearrange_list(input_list):
        half_length = len(input_list) // 2
        rearranged_list = input_list[:2] + input_list[1::2]
        return rearranged_list
    print(f"Минимального по модулю элемент списка: {int(m_value)}")
    print(f"Сумма модулей элементов после первого элемента равного нулю: {summ}")

```

```
print(f'Преобразованный список: {rearrange_list(input_list)}')
```

```
Введите список целых чисел через пробел  
-4 -2 1 12 -3 21 8 2 -22  
Минимального по модулю элемент списка: 1  
Сумма модулей элементов после первого элемента равного нулю: 0  
Преобразованный список: [-4, 1, -3, 8, -22, -2, 12, 21, 2]
```

Рисунок 5. Вывод программы индивидуального задания 2

Ответы на контрольные вопросы:

1. Что такое списки в языке Python?

Список (list) – это структура данных для хранения объектов различных типов.

2. Как осуществляется создание списка в Python?

Для создания списка нужно заключить элементы в квадратные скобки.

3. Как организовано хранение списков в оперативной памяти?

При создании списка в памяти резервируется область, которую можно условно назвать некоторым «контейнером», в котором хранятся ссылки на другие элементы данных в памяти. В отличие от таких типов данных как число или строка, содержимое «контейнера» списка можно менять.

4. Каким образом можно перебрать все элементы списка?

Можно воспользоваться циклом for: for i in list.

5. Какие существуют арифметические операции со списками?

Для объединения списков можно использовать оператор сложения (+).

Список можно повторить с помощью оператора умножения (*).

6. Как проверить есть ли элемент в списке?

Для того, чтобы проверить, есть ли заданный элемент в списке Python необходимо использовать оператор in.

7. Как определить число вхождений заданного элемента в списке?

Метод `count(элемент)` можно использовать для определения числа сколько раз данный элемент встречается в списке.

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

Метод `insert(индекс вставки, элемент)` можно использовать, чтобы вставить элемент в список.

Метод `append()` можно использовать для добавления элемента в конец списка.

9. Как выполнить сортировку списка?

Для сортировки списка нужно использовать метод `sort()`.

10. Как удалить один или несколько элементов из списка? Удалить элемент можно, написав его индекс в методе `pop(индекс)`. Элемент можно удалить с помощью метода `remove(значение)`. Оператор `del` можно использовать для тех же целей.

Можно удалить все элементы из списка с помощью метода `clear`.

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

List Comprehensions чаще всего на русский язык переводят как абстракция списков или списковое включение, является частью синтаксиса языка, которая предоставляет простой способ построения списков. В языке Python есть две очень мощные функции для работы с коллекциями: `map` и `filter`. Они позволяют использовать функциональный стиль программирования, не прибегая к помощи циклов, для работы с такими типами как `list`, `tuple`, `set`, `dict` и т.п. Списковое включение позволяет обойтись без этих функций.

Примеры: `a = [i for i in range(n)]` создаст список `a`, содержащий все числа от 0 до `n` не включительно. `b = list(map(lambda x: x**2, a))` создаст список `b`, в

котором каждый элемент будет являться квадратом соответствующего элемента из списка `a`. `b = list(filter(lambda x: x % 2 == 0, a))` создаст список `b`, содержащий только четные элементы списка `a`.

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

Слайсы (срезы) являются очень мощной составляющей Python, которая позволяет быстро и лаконично решать задачи выборки элементов из списка. Слайс задается тройкой чисел, разделенных запятой: `start:stop:step`. `Start` – позиция, с которой нужно начать выборку, `stop` – конечная позиция, `step` – шаг. При этом необходимо помнить, что выборка не включает элемент определяемый `stop`.

13. Какие существуют функции агрегации для работы со списками? Для работы со списками Python предоставляет следующие функции: `len(L)` – получить число элементов в списке `L`.

`min(L)` - получить минимальный элемент списка `L`. `max(L)` - получить максимальный элемент списка `L`.

`sum(L)` - получить сумму элементов списка `L`, если список `L` содержит только числовые значения.

Для функций `min` и `max` элементы списка должны быть сравнимы между собой.

14. Как создать копию списка?

Для создания копии списка необходимо использовать либо метод `copy()`, либо использовать оператор среза.

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

Отличие между `sorted()` и `sort()` заключается в том, что `sorted(список)`

возвращает новый отсортированный список без изменения исходного, а `sort()` изменяет сам исходный список.

Вывод: в результате выполнения работы были приобретены навыки по работе со списками при написании программ с помощью языка программирования Python версии 3.x.