

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 «Программирование на Python»**

Выполнил:  
Середа Кирилл Витальевич  
1 курс, группа ИВТ-б-о-22-1,  
09.03.01 «Информатика и  
вычислительная техника», очная  
форма обучения

---

(подпись)

Руководитель практики:  
Воронкин Роман Александрович

---

(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.

Ход выполнения:

1) Пример 1 - Ввести список A из 10 элементов, найти сумму элементов, меньших по модулю 5, и вывести ее на экран.



1 1 1 1 1 1 3 4 1 1  
15

Рисунок 1 Результат выполнения программы

Код программы:

```
#!/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)
```

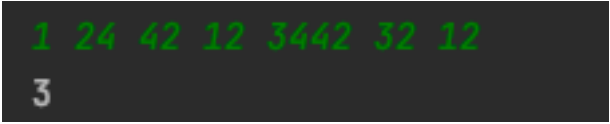
Альтернативная версия программы с тем же функционалом:

```
#!/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 = sum([a for a in A if abs(a) < 5])
    print(s)
```

2) Пример 2 - Написать программу, которая для целочисленного списка определяет, сколько положительных элементов располагается между его максимальным и минимальным элементами.



```
1 24 42 12 3442 32 12
3
```

Рисунок 2 – Результат работы программы

Код программы:

```
#!/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)
```

3) Индивидуальное задание 1 – Вариант 1 - Ввести список A из 10 элементов, найти наибольший элемент и переставить его с первым элементом. Преобразованный массив вывести.

```
Введите элемент 1: 2
Введите элемент 2: 1
Введите элемент 3: 3
Введите элемент 4: 4
Введите элемент 5: 5
Введите элемент 6: 6
Введите элемент 7: 10
Введите элемент 8: 5
Введите элемент 9: 4
Введите элемент 10: 3
Преобразованный массив: [10, 1, 3, 4, 5, 6, 2, 5, 4, 3]
```

Рисунок 3 – Результат выполнения программы

Код программы с использованием циклов:

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

if __name__ == '__main__':

    A = []
    for i in range(10):
        element = int(input(f"Введите элемент {i + 1}: "))
        A.append(element)

    max_element = max(A)
    max_index = A.index(max_element)

    A[0], A[max_index] = A[max_index], A[0]

    print("Преобразованный массив:", A)
```

Код программы с использованием List Comprehensions:

```
A = [int(input(f"Введите элемент {i + 1}: ")) for i in range(10)]

max_element = max(A)

A_transformed = [max_element] + [x for x in A if x != max_element]

print("Преобразованный массив:", A_transformed)
```

4) Индивидуальное задание 2 – Вариант 1 - В списке, состоящем из вещественных элементов, вычислить:

- сумму отрицательных элементов списка;
- произведение элементов списка, расположенных между максимальным и минимальным элементами.

```
Введите количество элементов списка: 6
Введите элемент 1: -1
Введите элемент 2: -4
Введите элемент 3: 2
Введите элемент 4: 3
Введите элемент 5: 4
Введите элемент 6: 2324
Сумма отрицательных элементов: -5.0
Произведение элементов между максимальным и минимальным: 24.0
```

Рисунок 4 – Результат работы программы

Код программы:

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

if __name__ == '__main__':

    nums_list = []
    n = int(input("Введите количество элементов списка: "))
    for i in range(n):
        item = float(input(f"Введите элемент {i + 1}: "))
        nums_list.append(item)

    negative_sum = sum([item for item in nums_list if item < 0])

    max_index = nums_list.index(max(nums_list))
    min_index = nums_list.index(min(nums_list))

    if max_index < min_index:
        max_index, min_index = min_index, max_index

    mult = 1
    for item in nums_list[min_index + 1:max_index]:
        mult *= item

    print("Сумма отрицательных элементов:", negative_sum)
    print("Произведение элементов между максимальным и минимальным:", mult)
```

Ответы на вопросы:

1. Что такое списки в языке Python? Список – это структура данных для хранения объектов различных типов. Размер списка не статичен, его можно изменять. Список по своей природе является изменяемым типом данных. Переменная, определяемая как список, содержит ссылку на структуру в памяти, которая в свою очередь хранит ссылки на какие-либо другие объекты или структуры.

2. Как осуществляется создание списка в Python? Для создания списка нужно заключить элементы в квадратные скобки: `list_1 = [1, 2, 3, 4,]`. Также при помощи `list()`: `list_2 = list(1, 2, 3, 4)`.

3. Как организовано хранение списков в оперативной памяти? При его создании в памяти резервируется область («контейнер»), в котором хранятся ссылки на другие элементы в памяти. Содержимое контейнера можно менять в отличие от чисел или строк.

4. Каким образом можно перебрать все элементы списка? Все элементы списка можно перебрать разными способами, при помощи цикла `for` и с помощью `range(len(list))`: `for i in range(len(list)): ...` или при помощи также цикла `for` и с помощью `enumerate(a)`: `for i, item in enumerate(a): ....`

5. Какие существуют арифметические операции со списками? Объединение списков при помощи оператора сложения («+») и операция повторения при помощи оператора умножения («\*»).

6. Как проверить есть ли элемент в списке? Проверить есть ли заданный элемент в списке может оператор `in`, если нет заданного элемента `not in`.

7. Как определить число вхождений заданного элемента в списке? Число вхождений заданного элемента в списке может определить метод `count`, который в качестве аргумента принимает искомый элемент.

8. Как осуществляется добавление (вставка) элемента в список? Добавление элемента в список может осуществляться при помощи метода `append(a)`, который добавляет элемент `a` в конец списка, также можно добавить больше одного элемента методом `extend(a)`.

9. Как выполнить сортировку списка? Отсортировать массив можно при помощи метода `sort()`, чтобы отсортировать по «убыванию», `sort(reverse=True)`.

10. Как удалить один или несколько элементов из списка? Удалить элемент по его индексу может метод `pop(i)`. Удалить элемент по его

значению может метод `remove()`. Оператор `del` может удалять также как метод `remove`, но сразу несколько элементов: `del list[1:3]`.

11. Что такое списковое включение и как с его помощью осуществлять обработку списков? Списковое включение – способ построения списков, пример: `a = [i for i in range(7)]`. В этом примере создастся массив из 6 элементов: от 0 до 6. Также при помощи данного способа можно осуществлять обработку списков: с элементом `i` проводить арифметические операции, и после цикла писать условие вхождения в список.

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

1. `list[:]` – копия списка;
2. `list[0:n]` – первые `n` элементы;
3. `list[n:m]` – получить элементы с `n+1` по `m`;
4. `list[::n]` – взять элементы списка с шагом `n`;
5. `list[n:m:s]` – взять элементы с `n+1` по `m` с шагом `s`.

13. Какие существуют функции агрегации для работы со списками? Функции агрегации: получить число элементов: `len()`, получить минимальный элемент списка: `min()`, получить максимальный элемент списка: `max()`, получить сумму элементов списка: `sum()`.

14. Как создать копию списка? Создать копию списка можно при помощи среза `a = b[:]` и при помощи метода `copy()`.

15. Самостоятельно изучите функцию `sorted` языка Python. В чем ее отличие от метода `sort` списков? Функция `sorted()` возвращает новый отсортированный список, оставляя исходный список неизменным. Она принимает список (или другую итерируемую последовательность) в качестве аргумента и возвращает новый список, содержащий отсортированные элементы. Основное отличие между `sorted()` и `sort()` заключается в том, что `sorted()` возвращает новый отсортированный список, оставляя исходный список неизменным, в то время как `sort()` изменяет сам список, сортируя его элементы на месте

Вывод: в ходе выполнения лабораторной работы были изучены списки, а также методы работы с ними, в том числе циклы и List Comprehensions.