

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»**  
**Вариант 29**

Выполнил:  
Саенко Андрей Максимович  
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», заочная  
форма обучения

---

(подпись)

Руководитель практики:  
Воронкин Р.А., канд. технических  
наук, 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. Проработаны примеры из лабораторной работы.

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

Результат работы программы:

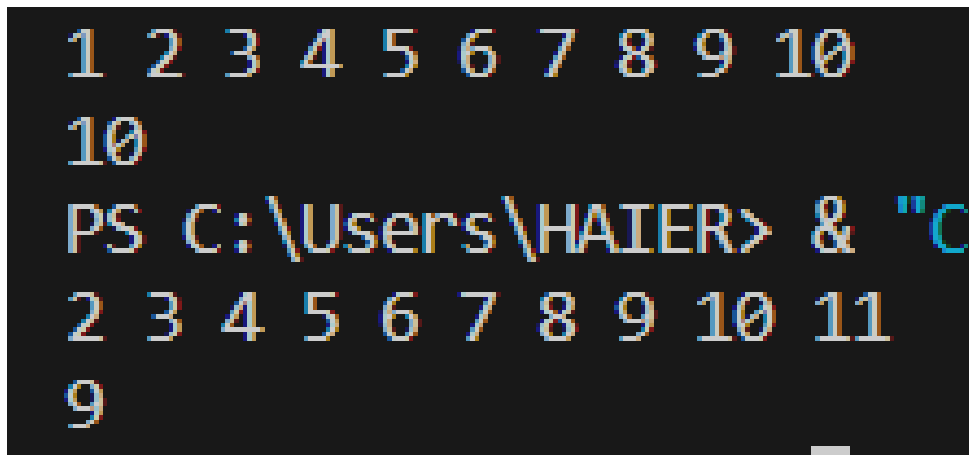


Рисунок 1 – Результат работы программы

Код программы:

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

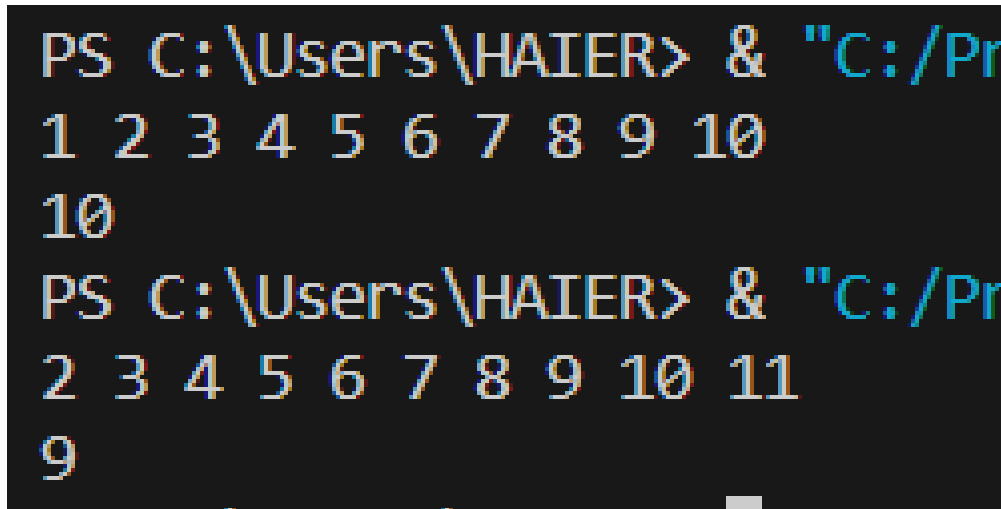
import sys

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

Второй способ решения задачи:

Результат работы программы:



```
PS C:\Users\HAIER> & "C:/Pr
1 2 3 4 5 6 7 8 9 10
10
PS C:\Users\HAIER> & "C:/Pr
2 3 4 5 6 7 8 9 10 11
9
```

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()))
    # 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)
```

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

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

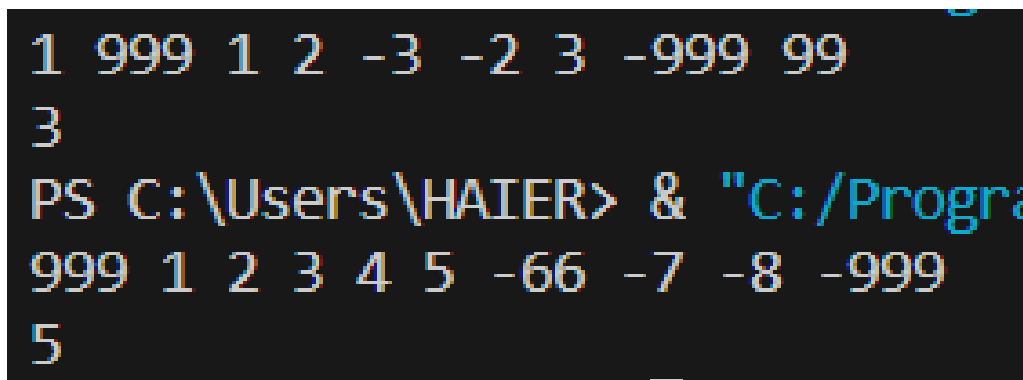
    # Проверить индексы и обменять их местами.
    if i_min > i_max:
        i_min, i_max = i_max, i_min

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

    print(count)

```

Результат работы программы:



```

1 999 1 2 -3 -2 3 -999 99
3
PS C:\Users\HAIER> & "C:/Program
999 1 2 3 4 5 -66 -7 -8 -999
5

```

Рисунок 3 – Результат работы программы

2. Выполнены индивидуальные задания

Вариант 29

Индивидуальное задание 1

Ввести список А из 10 элементов, найти наибольший элемент и переставить его с первым элементом. Преобразованный массив вывести.

Код программы:

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

import sys

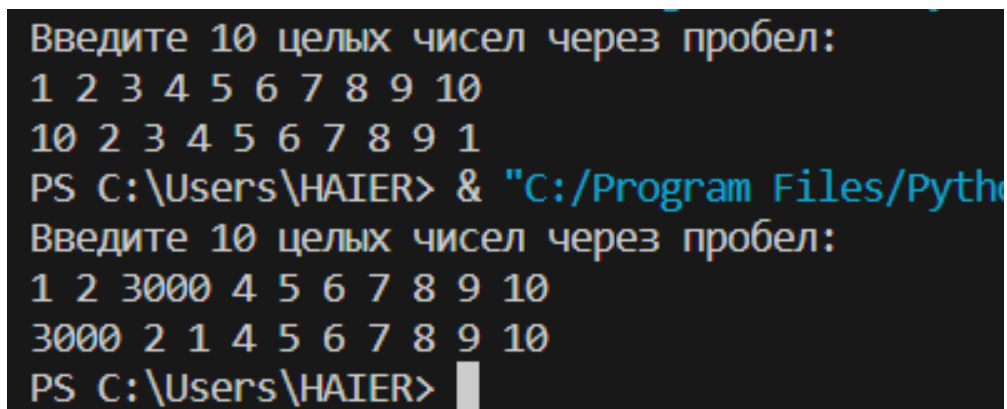
if __name__ == '__main__':
    A = [int(i) for i in input("Введите 10 целых чисел через пробел:\n").split()]
    if len(A) != 10:
        print("Должно быть введено 10 элементов", file=sys.stderr)
        exit(1)

    index_max = A.index(max(A))
    temp_first = A[0]
    temp_second = A[index_max]

    A[0] = temp_second
    A[index_max] = temp_first

    for item in A:
        print(item, end=' ')
```

Результат работы программы:



```
Введите 10 целых чисел через пробел:
1 2 3 4 5 6 7 8 9 10
10 2 3 4 5 6 7 8 9 1
PS C:\Users\HAIER> & "C:/Program Files/Python...
Введите 10 целых чисел через пробел:
1 2 3000 4 5 6 7 8 9 10
3000 2 1 4 5 6 7 8 9 10
PS C:\Users\HAIER>
```

Рисунок 4 – Результат работы программы

Решение задачи с использованием циклов:

Код программы:

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

import sys

if __name__ == '__main__':
    A = input().split()
```

```

if len(A) != 10:
    print("Должно быть введено 10 элементов", file=sys.stderr)
    exit(1)

B = []
for item in A:
    B.append(int(item))

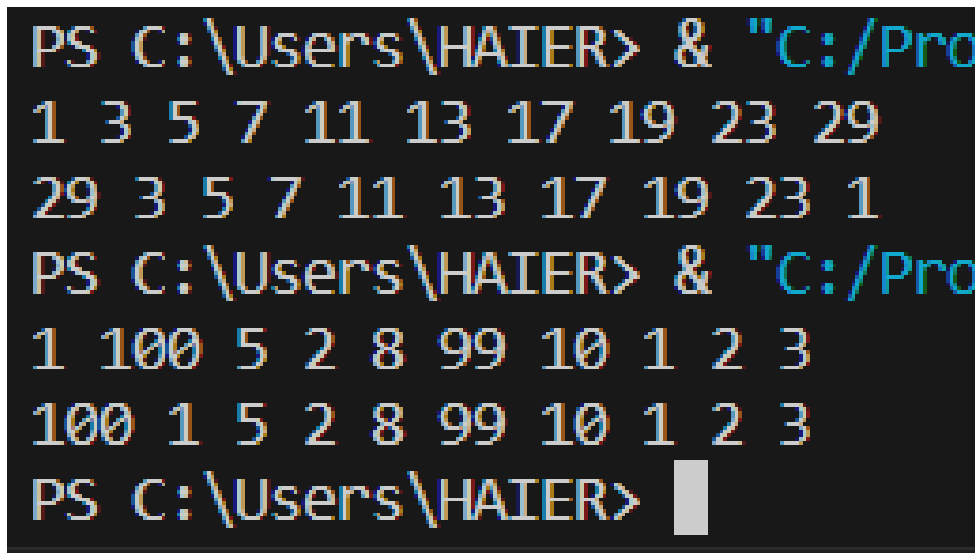
index_max = B.index(max(B))
temp_first = B[0]
temp_second = B[index_max]

B[0] = temp_second
B[index_max] = temp_first

for item in B:
    print(item, end=' ')

```

Результат работы программы:



```

PS C:\Users\HAIER> & "C:/Pro
1 3 5 7 11 13 17 19 23 29
29 3 5 7 11 13 17 19 23 1
PS C:\Users\HAIER> & "C:/Pro
1 100 5 2 8 99 10 1 2 3
100 1 5 2 8 99 10 1 2 3
PS C:\Users\HAIER>

```

Рисунок 5 – Результат работы программы

## Индивидуальное задание 2

В списке, состоящем из вещественных элементов, вычислить:

- 1) Номер минимального по модулю элемента списка;
- 2) Сумму модулей элементов списка, расположенных после первого отрицательного элемента.

Сжать список, удалив из него все элементы, величина которых находится в интервале  $[a, b]$ . Освободившиеся в конце списка элементы заполнить нулями.

### Код программы:

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

if __name__ == '__main__':
    A = [float(i) for i in input("Введите числа через пробел\n").split()]
    B = [abs(i) for i in A]
    index_min = B.index(min(B))
    print(f'Номер минимального по модулю элемента списка: {B.index(min(B))}')

    index_negative = 0
    counter_negative = 0
    for i, item in enumerate(A):
        if item < 0:
            index_negative = i
            counter_negative += 1
            break

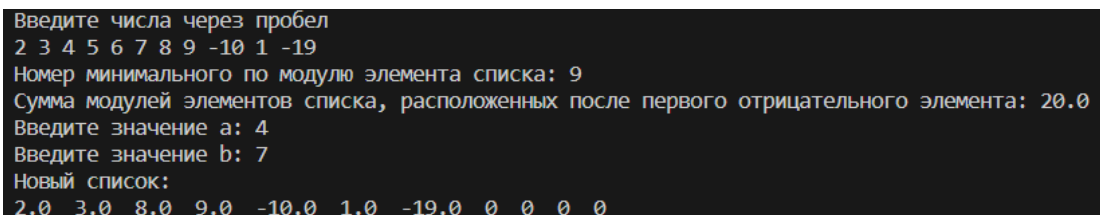
    sum_elements = 0
    if counter_negative == 0:
        print("В списке нет отрицательных элементов")
    else:
        sum_elements = sum(B[index_negative+1:])
        print(f'Сумма модулей элементов списка, расположенных после'
              f' первого отрицательного элемента: {sum_elements}')

    a = float(input("Введите значение a: "))
    b = float(input("Введите значение b: "))

    C = [i for i in A if i < a or i > b]
    del_counter = len(A) - len(C)

    print("Новый список:")
    print(*(C + [0]*del_counter), sep=" ")
```

### Результат работы программы:



```
Введите числа через пробел
2 3 4 5 6 7 8 9 -10 1 -19
Номер минимального по модулю элемента списка: 9
Сумма модулей элементов списка, расположенных после первого отрицательного элемента: 20.0
Введите значение a: 4
Введите значение b: 7
Новый список:
2.0 3.0 8.0 9.0 -10.0 1.0 -19.0 0 0 0 0
```

Рисунок 6 – Результат работы программы

## Ответы на контрольные вопросы

### 1. Что такое списки в языке Python?

Список – это структура данных для хранения объектов различных типов.

### 2. Как осуществляется создание списка в Python?

Для создания списка нужно присвоить переменной набор элементов, разделённых запятой, заключённый в квадратные скобки.

### 3. Как организовано хранение списков в оперативной памяти?

При создании списка в памяти резервируется область, которую можно условно назвать «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?

Чтоб перебрать все элементы списка, можно воспользоваться циклом `for` следующего вида:

```
for <название переменной> in <название списка>:  
    <действия>
```

### 5. Какие существуют арифметические операции со списками?

Существуют следующие арифметические операции со списками в языке программирования Python:

- 1) Сложение;
- 2) Умножение на целое число.

### 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?

Необходимо использовать функцию `list.count(<элемент>)`.

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

Для добавления элемента в список используется команда `list.append(<элемент>)`.



9. Как выполнить сортировку списка?

Для сортировки списка используется метод `list.sort()`.

10. Как удалить один или несколько элементов из списка?

Для удаления одного элемента можно использовать метод

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

Списковые включения (list comprehensions) – это удобный и компактный синтаксис для создания списков в Python. Они позволяют создавать новые списки, применяя выражение к каждому элементу итерируемого объекта, и, при необходимости, фильтровать элементы с помощью условий. Для обработки списка A с помощью списковых включений необходимо создать список B, в котором будет находиться результат обработки элементов списка A, а для этого при создании списка B нужно внутри квадратных скобок создать цикл, который будет обрабатывать элементы из A.

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

Рядом с именем переменной, содержащей список, создаются квадратные скобки, внутри которых задаются границы и шаг среза. В результате будет получен набор элементов списка, попадающих в этот срез.

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.

14. Как создать копию списка?

Для создания копии списка используется метод `list.copy()`.

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

Метод `list.sort()` предназначен для сортировки списка на месте, то есть он изменяет сам список, а `sorted()` возвращает новый отсортированный список, не изменяя исходный.

#### Вывод

Приобрёл навыки по работе со списками при написании программ с помощью языка программирования Python версии 3.x.