

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»**  
**Вариант \_\_\_\_**

Выполнил:  
Иващенко Олег Андреевич  
2 курс, группа ИВТ-б-о-22-1,  
09.03.02 «Информационные и  
вычислительные машины»,  
направленность (профиль)  
«Программное обеспечение средств  
вычислительной техники и  
автоматизированных систем»

---

(подпись)

Руководитель практики:  
Воронкин Роман Александрович,  
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

Таблица 1.1 – Код программы example\_1.py

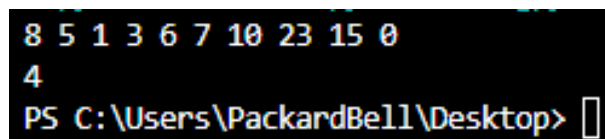
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

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)
```



```
8 5 1 3 6 7 10 23 15 0
4
PS C:\Users\PackardBell\Desktop>
```

Рисунок 1 – Вывод программы example\_1.py

Таблица 1.2 – Код программы example\_1.py

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

import sys

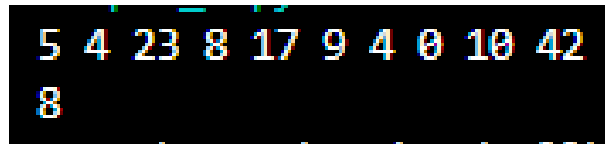
if __name__ == '__main__':
    # Ввести список одной строкой.
```

```

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)

```



```

5 4 23 8 17 9 4 0 10 42
8

```

Рисунок 1.2 – Вывод программы example\_2.py

Таблица 2 – Код программы example\_2.py

```

#!/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

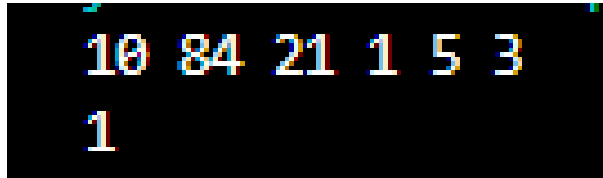
    # Проверить индексы и обменять их местами.
    if i_min > i_max:
        i_min, i_max = i_max, i_min

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

```

```
if item > 0:
    count += 1

print(count)
```



```
10 84 21 1 5 3
1
```

Рисунок 2 – Вывод программы example\_2.py

Индивидуальное задание 1. Ввести список A из 10 элементов, найти сумму элементов, меньших по модулю 3 и кратных 9, их количество и вывести результаты на экран.

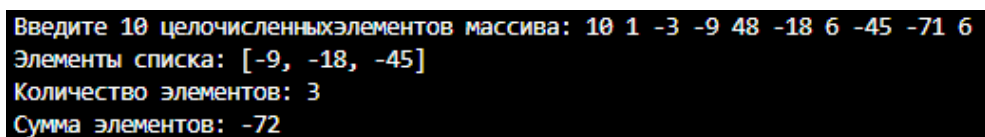
Таблица 3 – Код программы individual\_1.py

```
if __name__ == "__main__":
    A = list(map(int, input("Введите 10 целочисленных"
                           "элементов массива: ").split()))
    if (len(A) != 10):
        print("Количество элементов не равно 10. Завершение программы...")
        exit(1)

    result = []
    sum = 0

    for i in A:
        if i < abs(3) and i % 9 == 0:
            sum += i
            result.append(i)

    print(f"Элементы списка: {result}")
    print(f"Количество элементов: {len(result)}")
    print(f"Сумма элементов: {sum}")
```



```
Введите 10 целочисленныхэлементов массива: 10 1 -3 -9 48 -18 6 -45 -71 6
Элементы списка: [-9, -18, -45]
Количество элементов: 3
Сумма элементов: -72
```

Рисунок 3 – Вывод программы individual\_1.py

Индивидуальное задание 2. В списке, состоящем из вещественных элементов, вычислить:

1. Количество элементов списка, равных 0;
2. Сумму элементов списка, расположенных после минимального элемента.

Таблица 4 – Код программы individual\_2.py

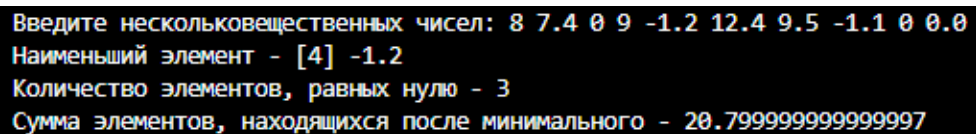
```
if __name__ == "__main__":
    A = list(map(float, input("Введите несколько"
                             "вещественных чисел: ").split()))

    i_min = 0
    min_sum = 0
    null_count = 0

    #Поиск минимального значения
    for i in A:
        if i < i_min:
            i_min = i
        if i == float(0):
            null_count += 1

    for i in range(A.index(i_min) + 1, len(A), 1):
        min_sum += A[i]

    print(f"Наименьший элемент - [{A.index(i_min)}] {i_min}")
    print(f"Количество элементов, равных нулю - {null_count}")
    print(f"Сумма элементов, находящихся после минимального - {min_sum}")
```



```
Введите нескольковещественных чисел: 8 7.4 0 9 -1.2 12.4 9.5 -1.1 0 0.0
Наименьший элемент - [4] -1.2
Количество элементов, равных нулю - 3
Сумма элементов, находящихся после минимального - 20.799999999999997
```

Рисунок 4 – Вывод программы individual\_2.py

### Контрольные вопросы

1. Что такое списки в языке Python?

Списки в Python – это упорядоченные изменяемые коллекции объектов. Они могут содержать элементы различных типов данных и позволяют обращаться к элементам по индексу.

2. Как осуществляется создание списка в Python?

Пример создания списка:

```
my_list = [1, 2, 3, "apple", "orange"]
```

3. Как организовано хранение списков в оперативной памяти?

Элементы списка хранятся в памяти последовательно, и каждый элемент имеет свой индекс.

4. Каким образом можно перебрать все элементы списка?

Перебрать все элементы списка можно с помощью циклов `for` и `for..in`

5. Какие существуют арифметические операции со списками?

Арифметические операции со списками:

- Сложение списков (конкатенация) – `result = list1 + list2`;
- Умножение списка – `result = list1 * 3`

6. Как проверить, есть ли элемент в списке?

Для проверки наличия элемента в списке можно использовать оператор `in`. Пример:

```
if element in list:
```

7. Как определить число вхождений заданного элемента в списке?

Для определения числа вхождений заданного элемента в списке можно использовать метод `count()`. Пример:

```
count = list.count(element)
```

8. Как осуществляется добавление (вставка) элементов в списке?

Добавление (вставка) элементов в список осуществляется методами `append()` или `insert()`.

9. Как выполнить сортировку списка?

Для выполнения сортировки списка можно использовать метод `sort()`.

10. Как удалить один или несколько элементов из списка?

Удаление элементов из списка можно осуществить с помощью метода `remove()`. Пример:

```
list.remove(element)
```

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

Создание нового списка с использованием выражения и цикла в одной строки можно осуществить следующим образом:

```
list = [x**2 for x in range(10)]
```

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

Доступ к элементам списков с помощью срезов можно осуществить следующим образом:

```
result = list[start_index:end_index]
```

13. Какие существуют функции агрегации для работы со списками?

Функции агрегации для работы со списками:

- `len(list)` – длина списка;
- `sum(list)` – сумма элементов списка;
- `max(list)` – максимальный элемент списка;
- `min(list)` – минимальный элемент списка.

14. Как создать копию списка?

Создание копии списка осуществляется методом `copy()`. Пример:

```
copy_list = list.copy()
```

15. Самостоятельно изучите функцию `sorted` языка Python. В чём отличие метода `sort` списков?

Функция `sorted()` создаёт новый отсортированный список. Метод `sort()` сортирует уже существующий список.

**Выводы:** В процессе выполнения лабораторной работы были приобретены навыки по работе со списками при написании программ с помощью языка программирования Python версии 3.x, были написаны 4 программы, 2 из которых являются примерами из лабораторной работы, и 2 индивидуальные задачи.