

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

Кафедра инфокоммуникаций

Основы кроссплатформенного программирования

**Отчет по лабораторной работе №5
Работа со списками в языке Python**

Выполнил студент группы

ИТС-б-з-22-1

Потеев А.С. « » _____ 2023г.

Подпись студента _____

Работа защищена « » _____ 2023г.

Проверил доцент, кандидат технических
наук, доцент кафедры инфокоммуникаций
Воронкин Роман Александрович

(подпись)

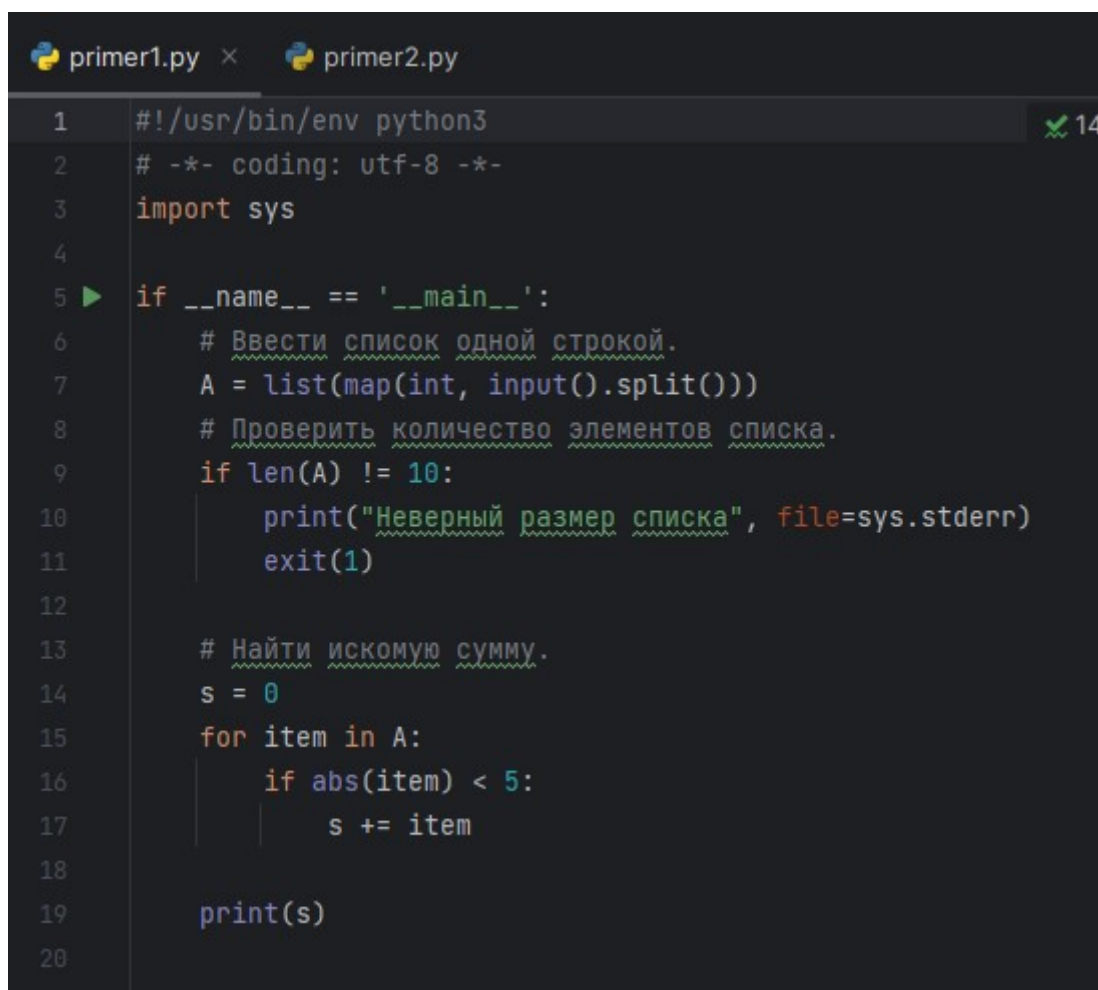
Ставрополь, 2023 г.

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

Ход работы:

Создал общедоступный репозиторий на GitHub (<https://github.com/poteevprog/Lab5>).

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

A screenshot of a code editor showing a Python script named 'primer1.py'. The script is written in a dark-themed editor with line numbers on the left. The code includes a shebang, encoding declaration, and imports. It defines a main function that takes user input, checks if the list has 10 elements, and calculates the sum of elements with an absolute value less than 5. The script ends with a print statement and an exit call.

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 import sys
4
5 if __name__ == '__main__':
6     # Ввести список одной строкой.
7     A = list(map(int, input().split()))
8     # Проверить количество элементов списка.
9     if len(A) != 10:
10         print("Неверный размер списка", file=sys.stderr)
11         exit(1)
12
13     # Найти искомую сумму.
14     s = 0
15     for item in A:
16         if abs(item) < 5:
17             s += item
18
19     print(s)
20
```

A screenshot of a terminal window showing the execution of the Python script. The prompt is 'C:\Users\user\Desktop\учеба\1 курс\2 семестр\Основы кро...'. The user has entered the list '1 2 3 1 2 3 1 2 3 9'. The output shows the sum '18'. The terminal also displays 'Process finished with exit code 0'.

```
"C:\Users\user\Desktop\учеба\1 курс\2 семестр\Основы кро...
1 2 3 1 2 3 1 2 3 9
18
Process finished with exit code 0
```

Рисунок 1 – Окно программы примера 1

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

```
primer1.py  primer2.py ×
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      # Ввести список одной строкой.
8      a = list(map(int, input().split()))
9      # Если список пуст, завершить программу.
10     if not a:
11         print("Заданный список пуст", file=sys.stderr)
12         exit(1)
13
14     # Определить индексы минимального и максимального элементов.
15     a_min = a_max = a[0]
16     i_min = i_max = 0
17     for i, item in enumerate(a):
18         if item < a_min:
19             i_min, a_min = i, item
20         if item >= a_max:
21             i_max, a_max = i, item
22
23     # Проверить индексы и обменять их местами.
24     if i_min > i_max:
25         i_min, i_max = i_max, i_min
26
27     # Посчитать количество положительных элементов.
28     count = 0
29     for item in a[i_min + 1:i_max]:
30         if item > 0:
31             count += 1
32
33     print(count)
34
```

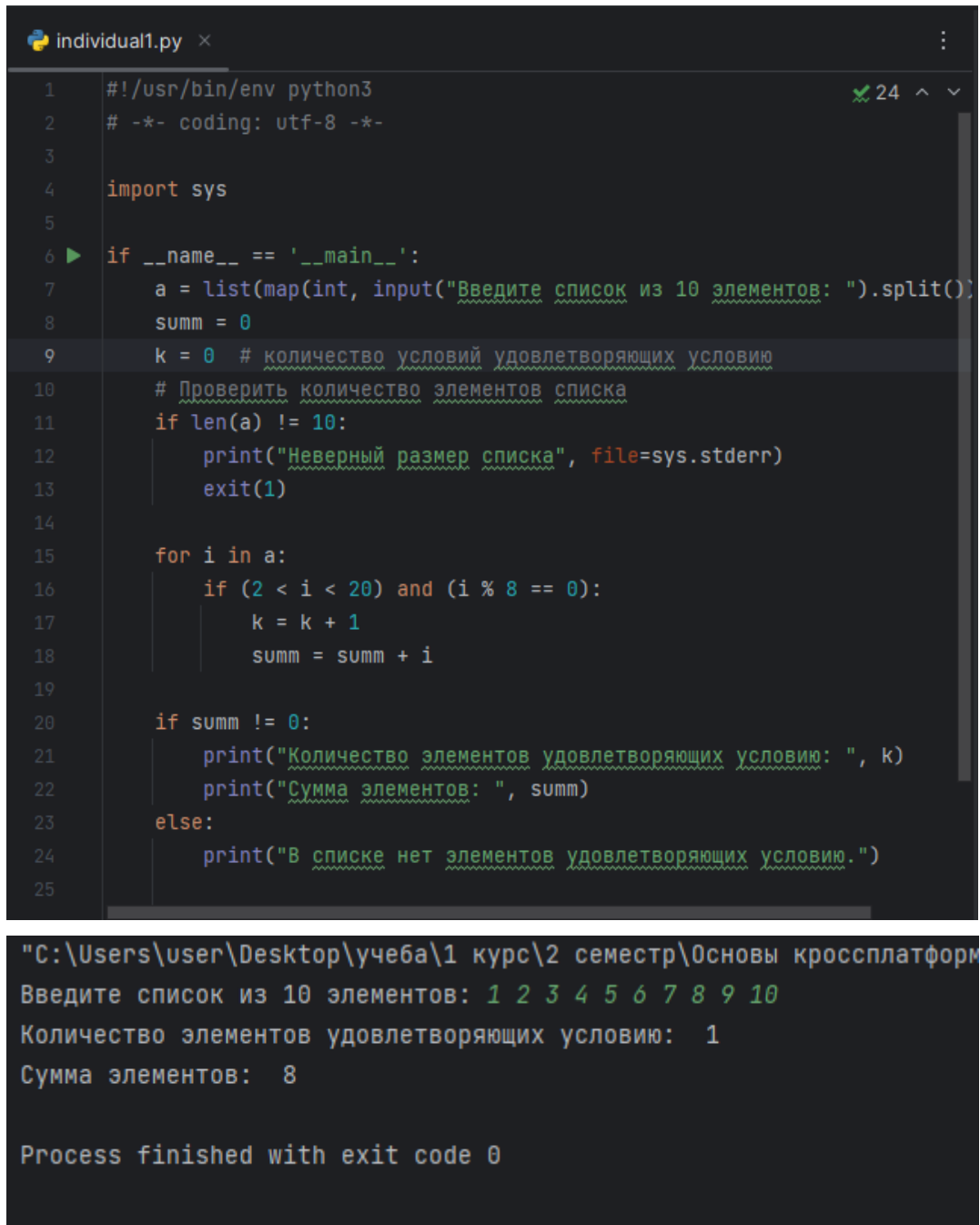
```
"C:\Users\user\Desktop\учеба\1 курс\2 семестр\0
1 2 3 1 2 3 7 44 67 45 43 3
7

Process finished with exit code 0
|
```

Рисунок 2 – Окно программы примера 2

Индивидуальное задание 1.

12. Ввести список A из 10 элементов, найти сумму элементов, больших 2 и меньших 20 и кратных 8, их количество и вывести результаты на экран.



```
individual1.py x
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      a = list(map(int, input("Введите список из 10 элементов: ").split()))
8      summ = 0
9      k = 0 # количество условий удовлетворяющих условию
10     # Проверить количество элементов списка
11     if len(a) != 10:
12         print("Неверный размер списка", file=sys.stderr)
13         exit(1)
14
15     for i in a:
16         if (2 < i < 20) and (i % 8 == 0):
17             k = k + 1
18             summ = summ + i
19
20     if summ != 0:
21         print("Количество элементов удовлетворяющих условию: ", k)
22         print("Сумма элементов: ", summ)
23     else:
24         print("В списке нет элементов удовлетворяющих условию.")
25
```

"C:\Users\user\Desktop\учеба\1 курс\2 семестр\Основы кроссплатформ

Введите список из 10 элементов: 1 2 3 4 5 6 7 8 9 10

Количество элементов удовлетворяющих условию: 1

Сумма элементов: 8

Process finished with exit code 0

Рисунок 3 – Окно программы для первой задачи и проверка кода на работоспособность.

Индивидуальное задание 2.

12. В списке, состоящем из вещественных элементов, вычислить:

1. количество элементов списка, лежащих в диапазоне от A до B ;
2. сумму элементов списка, расположенных после максимального элемента.

Упорядочить элементы списка по убыванию модулей элементов.

```
individual2.py ×
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      s = list(map(float, input("Введите список: ").split()))
8      a = float(input("Введите A: "))
9      b = float(input("Введите B: "))
10     summ = 0
11     k = 0 # Элементы лежащие в диапазоне от A до B
12     maxi = s[0] # Макс == значение первого элемента
13     j = 1
14     t = 0
15
16     if len(s) == 0:
17         print("Неверный размер списка!", file=sys.stderr)
18         exit(1)
19
20     for i in s:
21         if a < i < b:
22             k += 1
23
24         t = t + 1
25
26         if i > maxi:
27             maxi = i
28             j = t # индекс макс значения
29
30     # Сумма элементов после макс. значения
31     for i in range(len(s)):
32         if i > j - 1:
33             summ += s[i]
34
35     # Упорядочить элементы списка по возрастанию модуля
36     s1 = s
37     for i in range(len(s)):
38         s1[i] = abs(s1[i])
```

```
39
40     s1.sort()
41     s = s1
42
43     print("Упорядоченный список: ", s)
44     print("Количество элементов списка лежащих в диапазоне от A до B: ", k)
45     print("Сумма элементов списка, расположенных после максимального элемента: ", summ)
46
```

```
"C:\Users\user\Desktop\учеба\1 курс\2 семестр\Основы кроссплатформенного программи
Введите список: 1 2 3 4 5 6 7 8 9 10
Введите A: 2
Введите B: 4
Упорядоченный список: [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]
Количество элементов списка лежащих в диапазоне от A до B: 1
Сумма элементов списка, расположенных после максимального элемента: 0

Process finished with exit code 0
```

Рисунок 4 – Окно программы для второй задачи и проверка кода на работоспособность.

Ответы на контрольные вопросы:

1. Что такое списки в языке Python?

Список (list) – это структура данных для хранения объектов различных типов. В нем можно хранить объекты 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 на какие-либо другие объекты или структуры.

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? Читать элементы списка можно с помощью следующего цикла: `my_list = ['один', 'два', 'три', 'четыре', 'пять']`

```
for elem in my_list:  
    print(elem)
```

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` можно использовать для определения числа сколько раз данный элемент встречается в списке.

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

Метод `append` можно использовать для добавления элемента в список.

Метод `insert` можно использовать, чтобы вставить элемент в список.

9. Как выполнить сортировку списка?

Для сортировки списка нужно использовать метод `sort`. Для сортировки списка в порядке убывания необходимо вызвать метод `sort` с аргументом `reverse=True`.

10. Как удалить один или несколько элементов из списка?

Удалить элемент можно, написав его индекс в методе `pop`. Если не указывать индекс, то функция удалит последний элемент. Элемент можно удалить с помощью метода `remove`. Оператор `del` можно использовать для тех же целей.

Можно удалить несколько элементов с помощью оператора среза.

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.

12. Какие существуют функции агрегации для работы со списками?

Для работы со списками Python предоставляет следующие функции:

1. `len(L)` - получить число элементов в списке `L`
2. `min(L)` - получить минимальный элемент списка `L`
3. `max(L)` - получить максимальный элемент списка `L`
4. `sum(L)` - получить сумму элементов списка `L`, если список `L`

содержит только числовые значения.

13. Как создать копию списка? `copy.copy(x)`

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

Функция `sorted()` в Python возвращает отсортированный список из элементов в итерируемом объекте. `list.sort()` на 13% быстрее, чем `sorted()`.

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

Функция `sort()` очень похожа на `sorted()`, но в отличие от `sorted` она ничего не возвращает и не вносит изменений в исходную последовательность. Более того, `sort()` является методом класса `list` и может использоваться только со списками. Синтаксис: `List_name.sort(key, reverse=False)` Параметры: ключ: Функция, которая служит ключом для

сравнения **сортировки**. реверс: Если true, то **список** сортируется в порядке убывания.

Вывод: Я на лабораторной работе приобрел навыки по работе со списками при написании программ с помощью языка программирования Python.