

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 «Основы кроссплатформенного программирования»

Вариант ____

Выполнила:
Маньшина Дарья Алексеевна
1 курс, группа ИТС-б-о-22-1,
11.03.02 «Инфокоммуникационные
технологии и системы связи»,
направленность (профиль)
«Инфокоммуникационные системы и
сети», очная форма обучения

(подпись)

Руководитель практики:
Воронкин Р. А., канд. тех. наук,
доцент кафедры
инфокоммуникаций

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

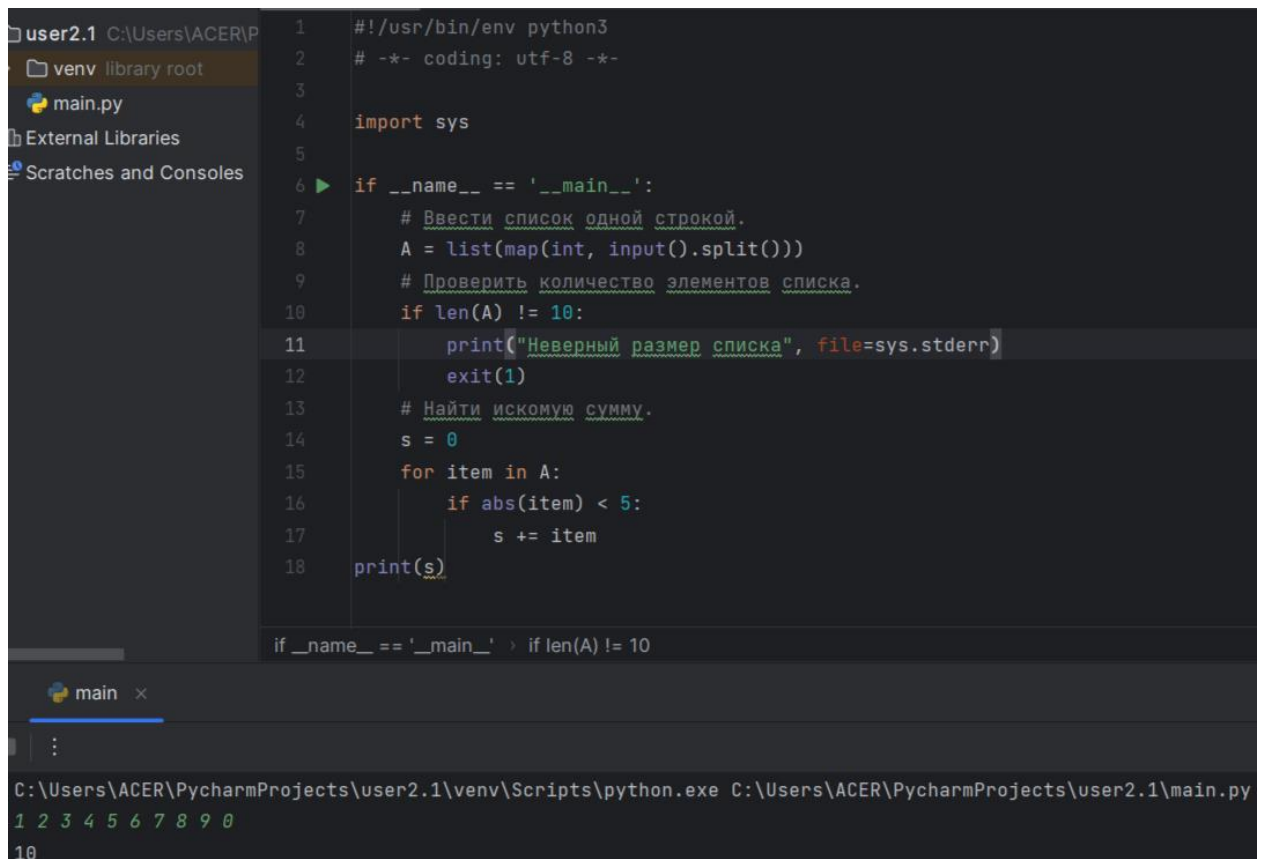
Тема: работа со списками в языке Python

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

Ход работы:

1. Выполним примеры данные в методичке.

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



```
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      # Проверить количество элементов списка.
10     if len(A) != 10:
11         print("Неверный размер списка", file=sys.stderr)
12         exit(1)
13     # Найти искомую сумму.
14     s = 0
15     for item in A:
16         if abs(item) < 5:
17             s += item
18     print(s)
```

if __name__ == '__main__' > if len(A) != 10

main x

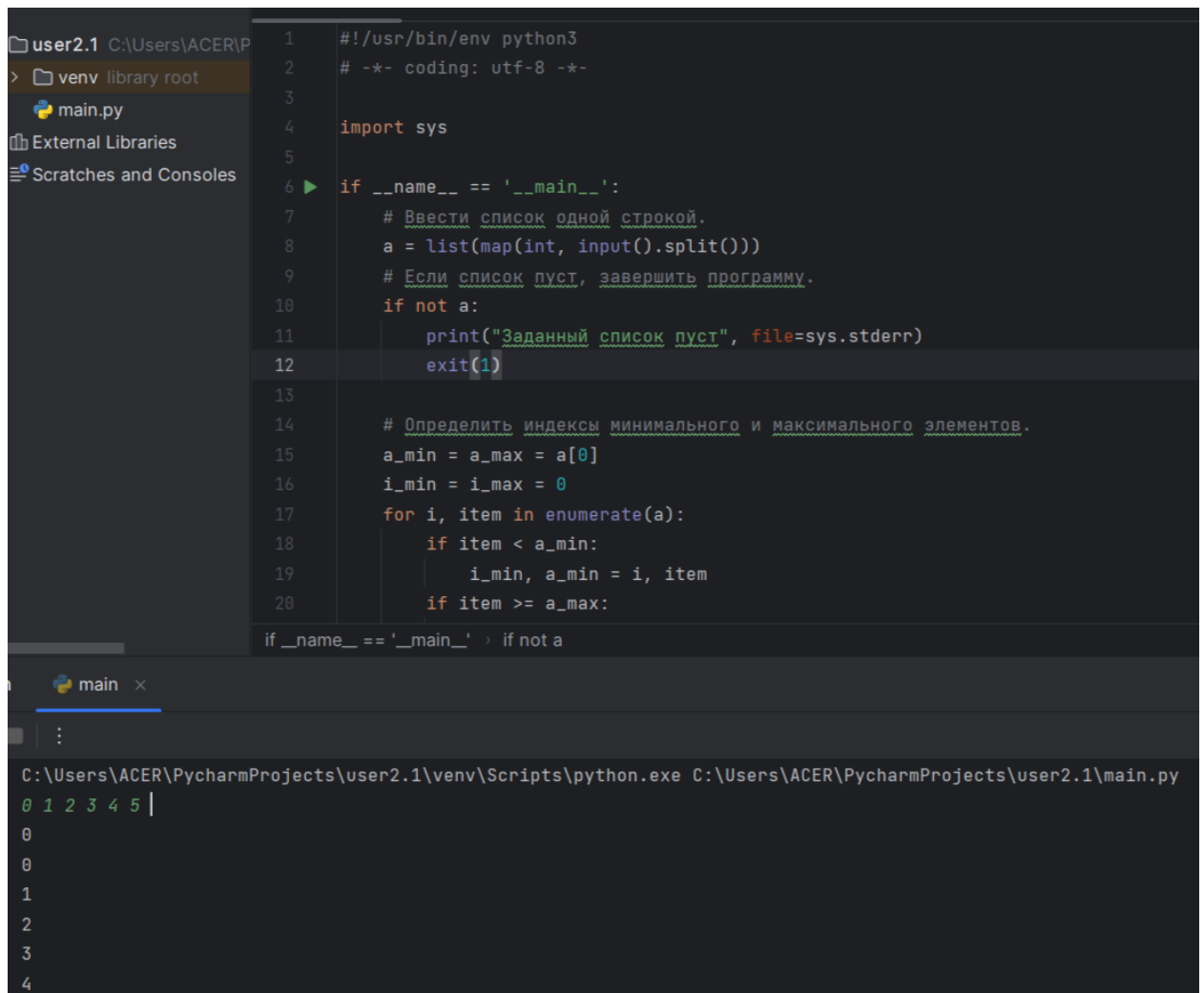
C:\Users\ACER\PycharmProjects\user2.1\venv\Scripts\python.exe C:\Users\ACER\PycharmProjects\user2.1\main.py

1 2 3 4 5 6 7 8 9 0

10

Рисунок 1 – Программа и решение для примера №1

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



```
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     if __name__ == '__main__': if not a
```

main x

C:\Users\ACER\PycharmProjects\user2.1\venv\Scripts\python.exe C:\Users\ACER\PycharmProjects\user2.1\main.py

0 1 2 3 4 5 |

0

0

1

2

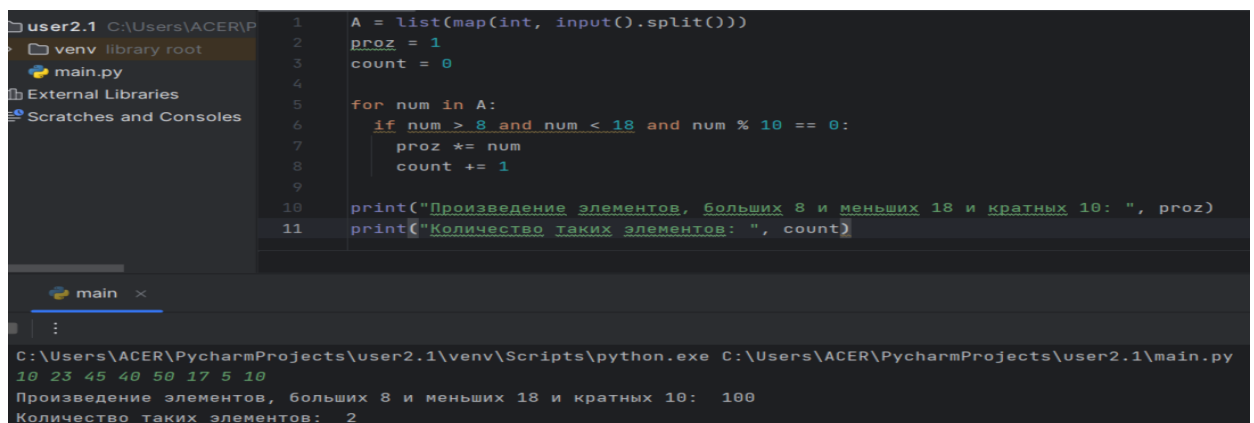
3

4

Рисунок 2 – Программа и решение для примера № 2

2. Решим индивидуальные задания. Вариант 15 (по списку группы).

Задание 1. Ввести список А из 10 элементов, найти произведение элементов, больших 8 и меньших 18 и кратных 10, их количество и вывести результаты на экран.



```
1  A = list(map(int, input().split()))
2  proz = 1
3  count = 0
4
5  for num in A:
6      if num > 8 and num < 18 and num % 10 == 0:
7          proz *= num
8          count += 1
9
10 print("Произведение элементов, больших 8 и меньших 18 и кратных 10: ", proz)
11 print("Количество таких элементов: ", count)
```

main x

C:\Users\ACER\PycharmProjects\user2.1\venv\Scripts\python.exe C:\Users\ACER\PycharmProjects\user2.1\main.py

10 23 45 40 50 17 5 10

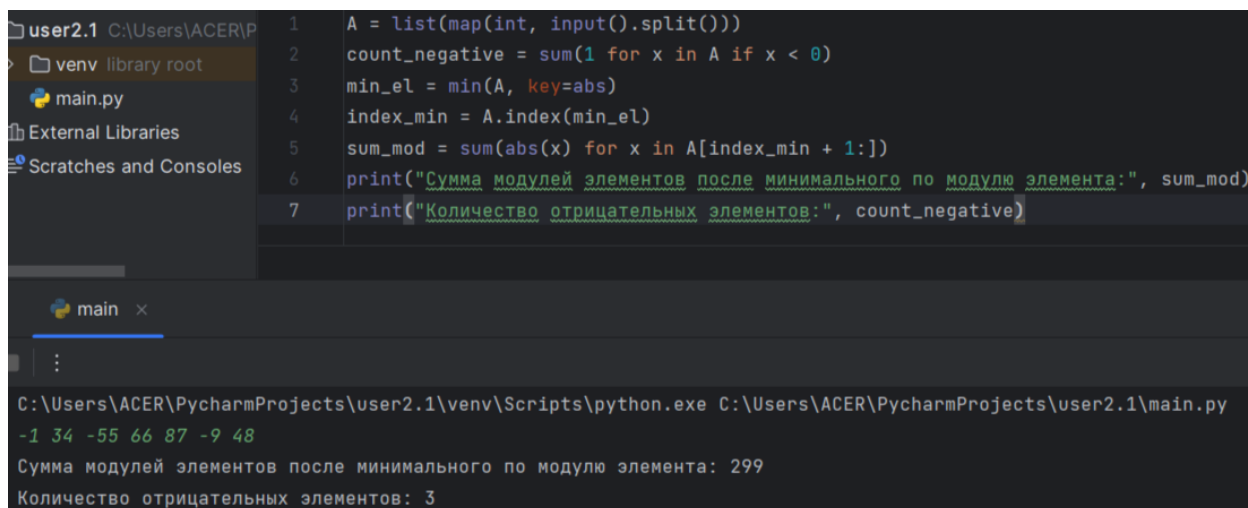
Произведение элементов, больших 8 и меньших 18 и кратных 10: 100

Количество таких элементов: 2

Рисунок 3 – Программа и решение для индивидуального задания №1

Задание 2. В списке, состоящем из вещественных элементов, ВЫЧИСЛИТЬ:

- 1) количество отрицательных элементов списка;
- 2) сумму модулей элементов списка, расположенных после минимального по модулю элемента.



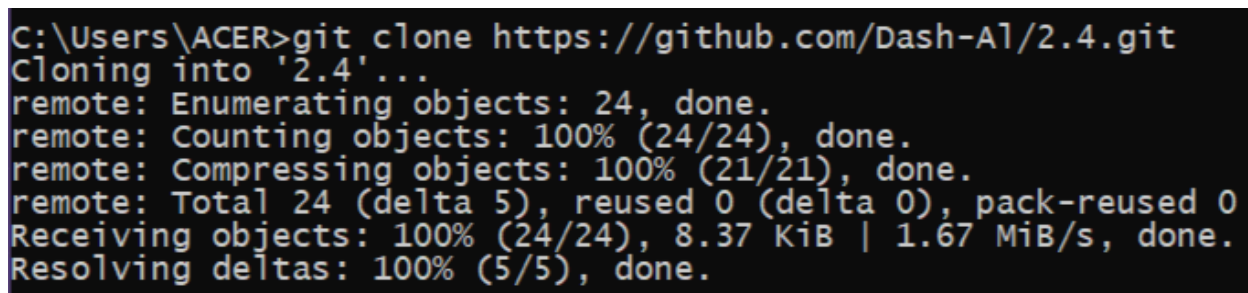
```
1 A = list(map(int, input().split()))
2 count_negative = sum(1 for x in A if x < 0)
3 min_el = min(A, key=abs)
4 index_min = A.index(min_el)
5 sum_mod = sum(abs(x) for x in A[index_min + 1:])
6 print("Сумма модулей элементов после минимального по модулю элемента:", sum_mod)
7 print("Количество отрицательных элементов:", count_negative)
```

main x

C:\Users\ACER\PycharmProjects\user2.1\venv\Scripts\python.exe C:\Users\ACER\PycharmProjects\user2.1\main.py
-1 34 -55 66 87 -9 48
Сумма модулей элементов после минимального по модулю элемента: 299
Количество отрицательных элементов: 3

Рисунок 4 – Программа и решение для индивидуального задания №2

3. Создала общедоступный репозиторий, в котором использована лицензия MIT и язык программирования Python. Был добавлен файл .gitignore. Клонирование репозитория. Организация репозитория с моделью ветвления git flow.



```
C:\Users\ACER>git clone https://github.com/Dash-A1/2.4.git
Cloning into '2.4'...
remote: Enumerating objects: 24, done.
remote: Counting objects: 100% (24/24), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 24 (delta 5), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (24/24), 8.37 KiB | 1.67 MiB/s, done.
Resolving deltas: 100% (5/5), done.
```

Рисунок 5 – Результат клонирования

```
C:\Git\2.4-main>git flow init
Initialized empty Git repository in C:/Git/2.4-main/.git/
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Bugfix branches? [bugfix/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? []
Hooks and filters directory? [C:/Git/2.4-main/.git/hooks]
```

Рисунок 6 – Результат команды git flow init

Ответы на контрольные вопросы

1. Что такое списки в языке Python?

Списки в Python – упорядоченный изменяемый набор объектов произвольных типов, пронумерованных от 0. Они используются для хранения и работы с данными.

2. Как осуществляется создание списка в Python?

Для **создания списка Python** нужно заключить элементы в квадратные скобки.

3. Как организовано хранение списков в оперативной памяти?

Списки в Python хранятся в оперативной памяти в виде последовательности элементов, каждый из которых может быть любого типа.

4. Каким образом можно перебрать все элементы списка?

Чтобы перебрать все элементы можно использовать цикл for

5. Какие существуют арифметические операции со списками?

+ – конкатенация списков; * – дублирование списка.

6. Как проверить есть ли элемент в списке?

Встроенный метод `count()` возвращает количество вхождений указанного элемента в списке. Если элемента в списке нет, то `count()` вернет 0. Если возвращается целое положительное число больше 0, это означает, что список содержит элемент.

Также можно использовать оператор `in`.

7. Как определить число вхождений заданного элемента в списке?

Метод `python count()` подсчитывает количество вхождений элемента в списке и возвращает найденное значение. Метод `count()` принимает один аргумент `x`, значение которое нужно найти. Данный метод возвращает количество вхождений элемента в список.

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

Добавление элемента в список можно осуществить при помощи метода `append()`

9. Как выполнить сортировку списка?

В Python данные можно сортировать с помощью методов `sorted()` или `sort()`.

10. Как удалить один или несколько элементов из списка?

Можно удалить элементы из списка, используя метод `remove()`

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

Списковое включение — это некий синтаксический сахар, позволяющий упростить генерацию последовательностей (списков,

множеств, словарей, генераторов). Функции для работы с коллекциями: `map` и `filter`.

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

Срезы позволяют обрезать список, взяв лишь те элементы, которые нужны. Доступ к элементам списка с помощью срезов осуществляется с помощью квадратных скобок `[]`.

13. Какие существуют функции агрегации для работы со списками?

`sum ()` – получить сумму элементов списка.

`min ()` – получение минимального элемента списка.

`max ()` – получение максимального элемента списка.

`len ()` – получить число элементов в списке.

14. Как создать копию списка?

Чтобы скопировать список, создайте срез, включающий весь исходный список без указания первого и последнего индекса `[:]`. Данная конструкция создаёт срез, который начинается с первого элемента и заканчивается последним, в результате этого создаётся копия списка.

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

Разница только в том, что **метод `sort`** не создает новой коллекции, а меняет уже существующую. **Функция же `sorted`** не меняет исходную коллекцию, а создает новую с отсортированными элементами.

Вывод: в ходе лабораторной работы приобрела навыки по работе со списками при написании программ с помощью языка программирования Python версии 3.x.

