

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»

Выполнил:
Евдаков Евгений Владимирович
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., 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:

Задание 1. Создал общедоступный репозиторий на GitHub, в котором использована лицензий MIT и язык программирования Python, также добавил файл .gitignore с необходимыми правилами. Клонировал свой репозиторий на свой компьютер.

```
C:\Users\Gaming-PC>git clone https://github.com/EvgenyEvdakov/Laba_2.4.git
Cloning into 'Laba_2.4'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (8/8), done.
Resolving deltas: 100% (1/1), done.
```

Рисунок 1. Клонирование репозитория

Задание 2. Организовал свой репозиторий в соответствии с моделью ветвления git-flow, появилась новая ветка develop в которой буду выполнять дальнейшие задачи.

```
C:\Users\Gaming-PC\Laba_2.4>git flow init
which branch should be used for bringing forth production releases?
- main
Branch name for production releases: [main]
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:/Users/Gaming-PC/Laba_2.4/.git/hooks]
```

Рисунок 2. Модель ветвления git-flow

Задание 3. Создал проект PyCharm в папке репозитория. Приступил к работе с примером №1. Добавил новый файл primer1.py.

Условие примера: ввести список A из 10 элементов, найти сумму элементов, меньших по модулю 5, и вывести ее на экран.

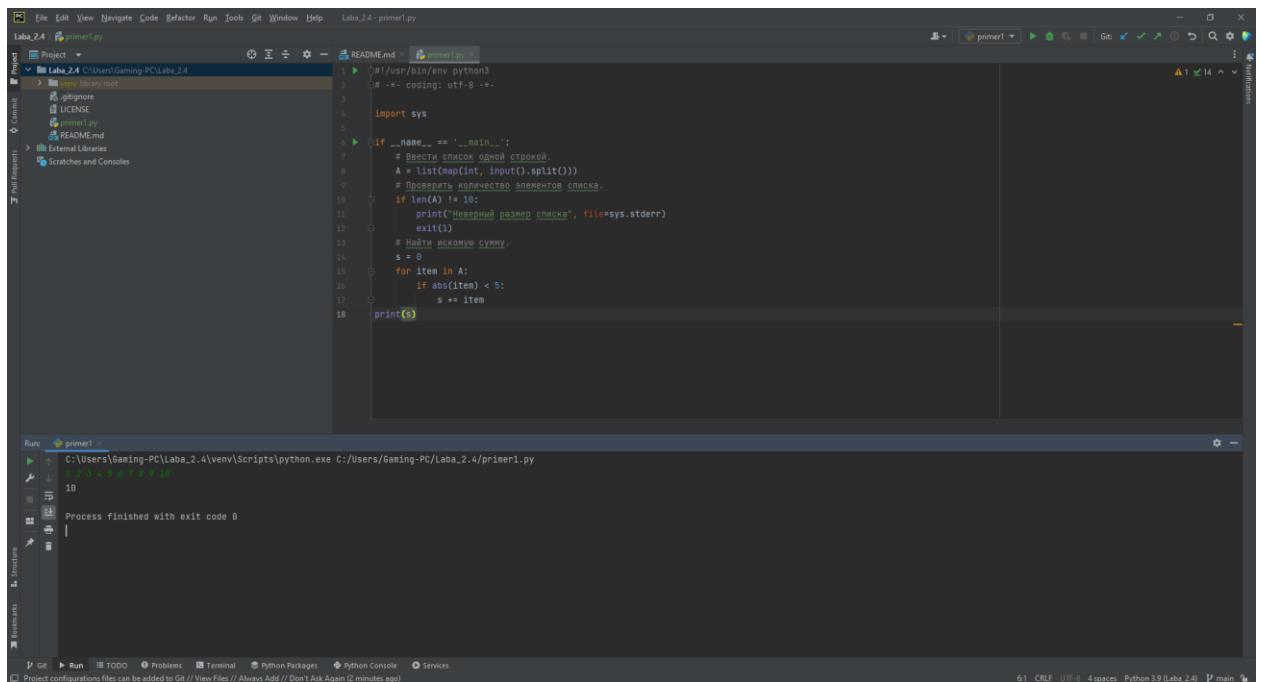


Рисунок 3. Реализация первого примера

Задание 4. Создал новый файл под названием primer2.py. Приступил к работе с примером №2.

Условие примера: написать программу, которая для целочисленного списка определяет, сколько положительных элементов располагается между его максимальным и минимальным элементами.

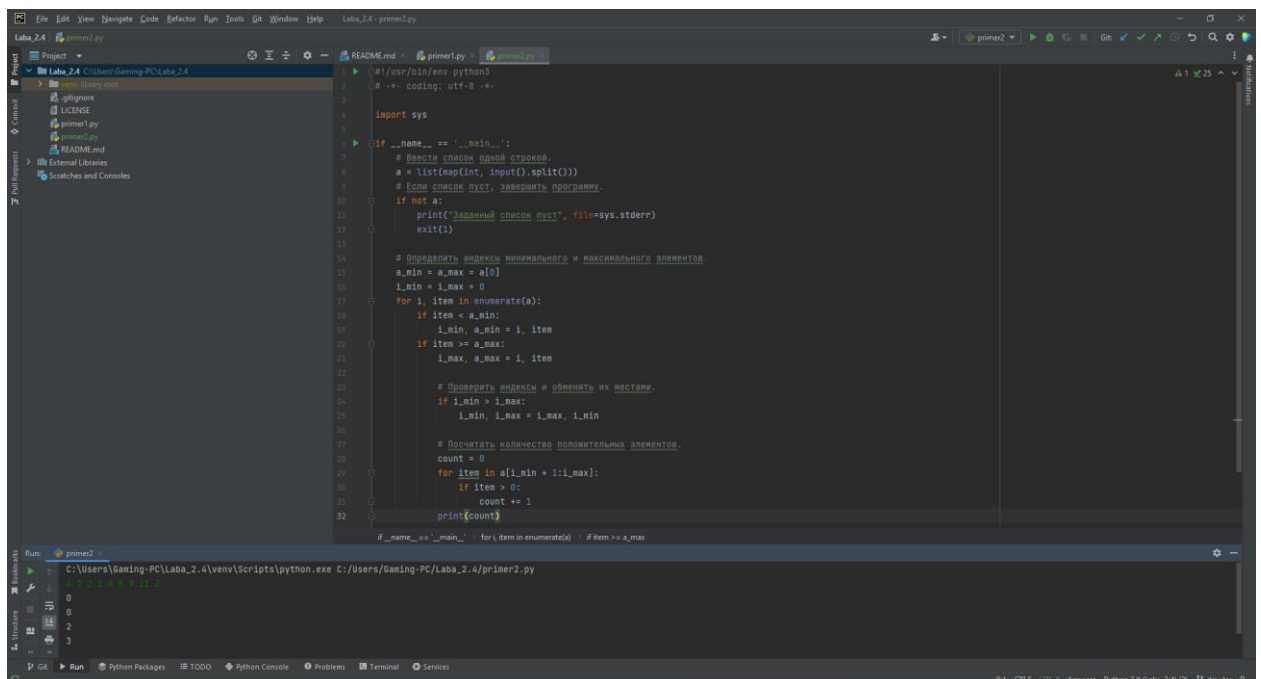


Рисунок 4. Реализация второго примера

Задание 5.

Индивидуальное задание 1

Вариант 9 (по списку группы)

Создал новый файл под названием individual1.py

Условие задания: Составить программу, выдающую индексы заданного элемента или сообщаящую, что такого элемента в списке нет.

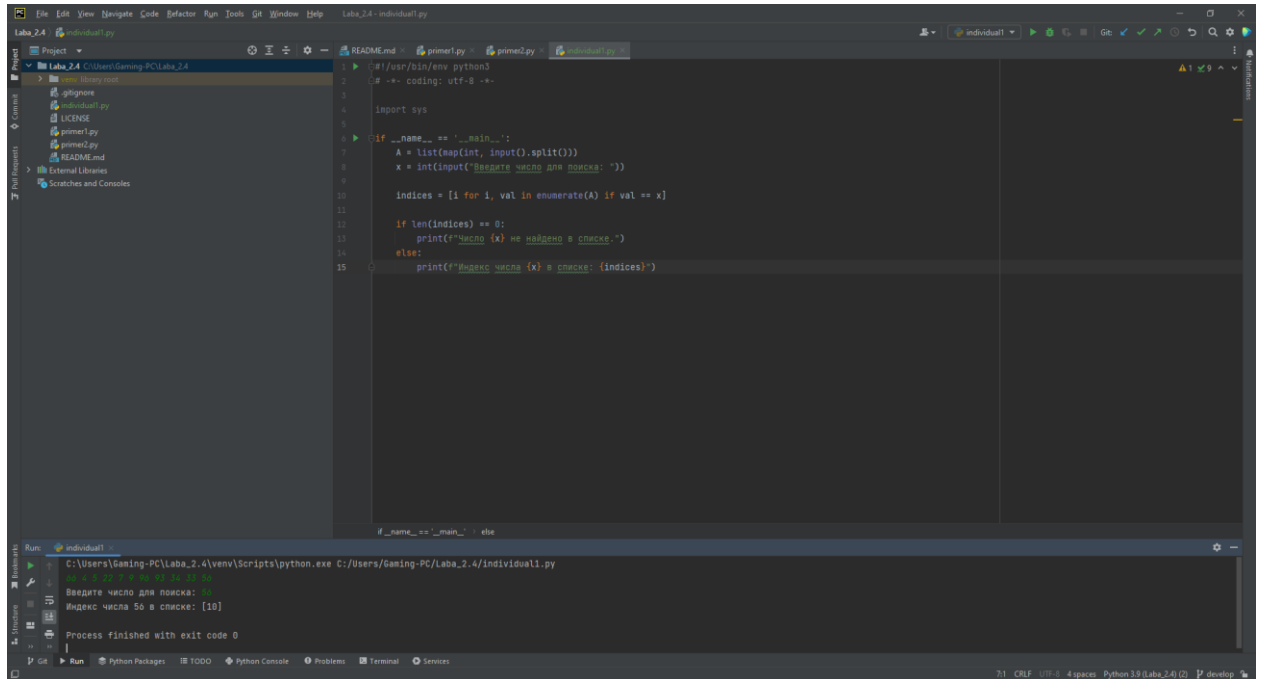


Рисунок 5. Программа с индексом заданного элемента

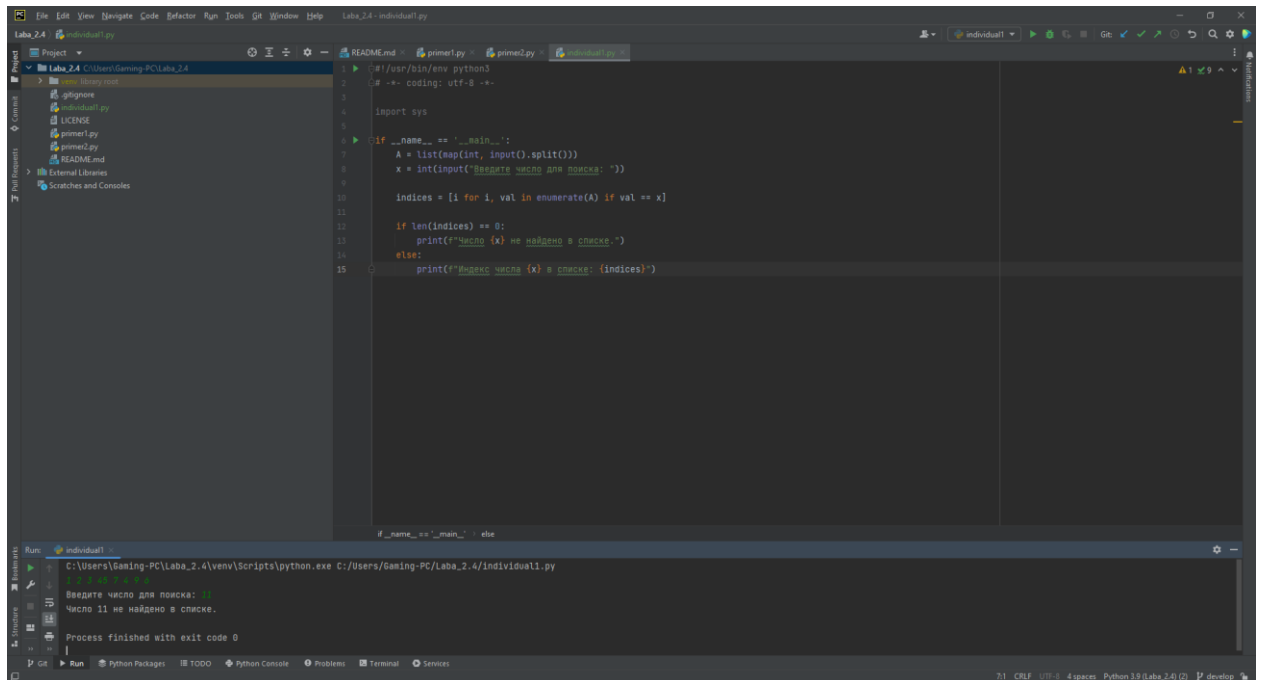


Рисунок 6. Программа с отсутствующим индексом заданного элемента

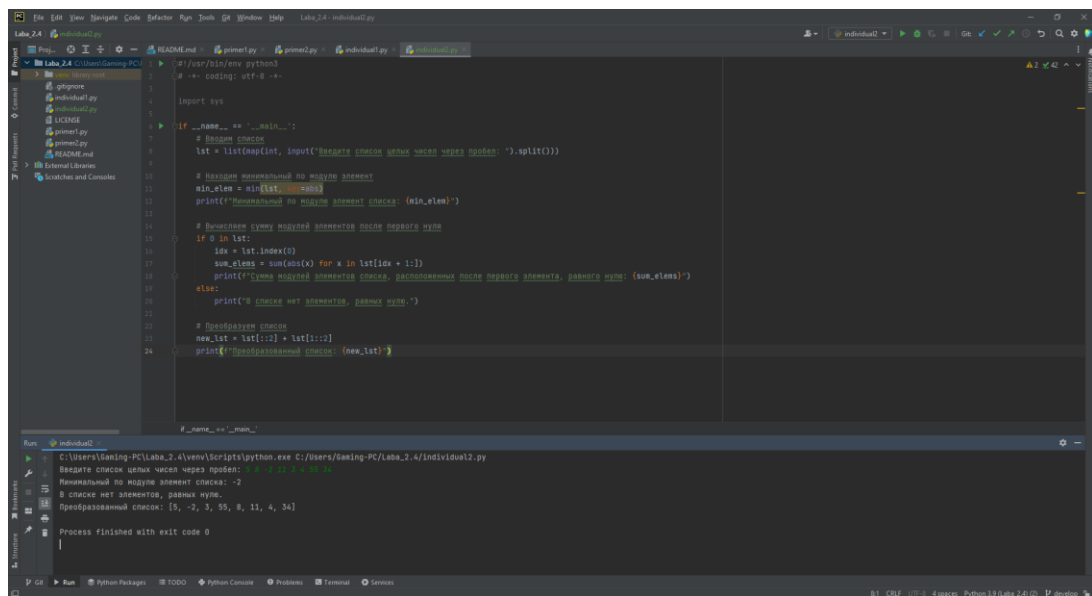
Индивидуальное задание 2

Вариант 9 (по списку группы)

Условие задания: в списке, состоящем из целых элементов, вычислить:

1. минимальный по модулю элемент списка;
2. сумму модулей элементов списка, расположенных после первого элемента, равного нулю.

Преобразовать список таким образом, чтобы в первой его половине располагались элементы, стоявшие в четных позициях, а во второй половине - элементы, стоявшие в нечетных позициях.



```
1 # -*- coding: utf-8 -*-
2
3 import sys
4
5 if __name__ == '__main__':
6     # Ввод списка
7     list = list(map(int, input('Введите список целых чисел через пробел: ').split()))
8
9     # Находим минимальный по модулю элемент
10    min_elem = min(list, key=abs)
11    print(f'Наименьший по модулю элемент списка: {min_elem}')
12
13    # Находим сумму модулей элементов после первого нуля
14    if 0 in list:
15        idx = list.index(0)
16        sum_elems = sum(abs(x) for x in list[idx + 1:])
17        print(f'Сумма модулей элементов списка, расположенных после первого элемента, равного нулю: {sum_elems}')
18    else:
19        print('В списке нет элементов, равных нулю.')
20
21    # Преобразуем список
22    new_list = list[::2] + list[1::2]
23    print(f'Преобразованный список: {new_list}')
24
25 if __name__ == '__main__':
```

Run: individual2

C:\Users\Gaming-PC\Lab2.4\venv\Scripts\python.exe C:\Users\Gaming-PC\Lab2.4\individual2.py

Введите список целых чисел через пробел: 5 -2 3 55 8 11 4 34

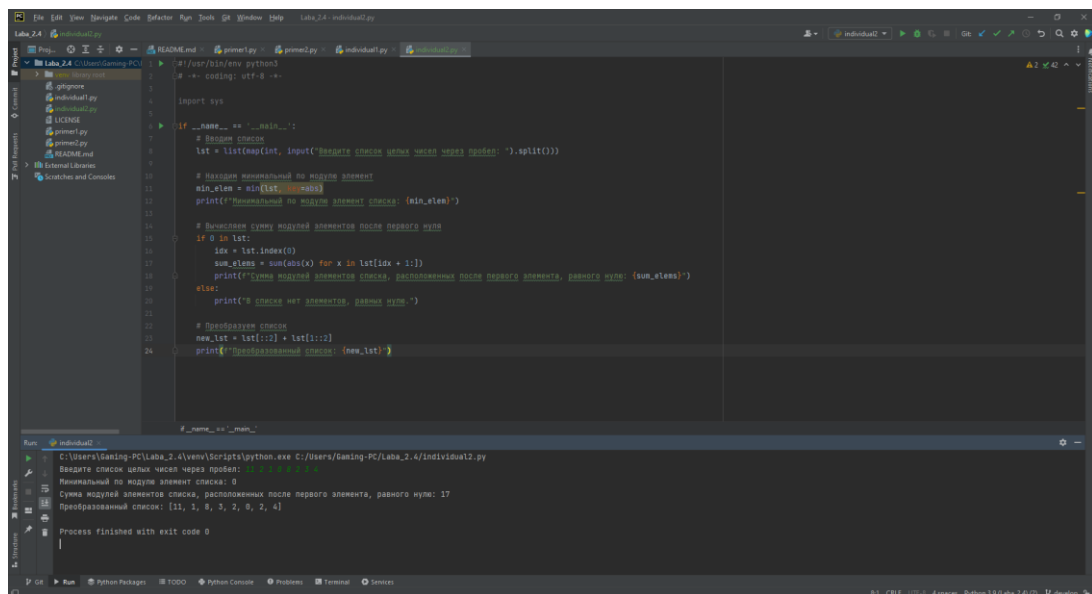
Наименьший по модулю элемент списка: -2

В списке нет элементов, равных нулю.

Преобразованный список: [5, -2, 3, 55, 8, 11, 4, 34]

Process finished with exit code 0

Рисунок 7. Программа без нулевого элемента



```
1 # -*- coding: utf-8 -*-
2
3 import sys
4
5 if __name__ == '__main__':
6     # Ввод списка
7     list = list(map(int, input('Введите список целых чисел через пробел: ').split()))
8
9     # Находим минимальный по модулю элемент
10    min_elem = min(list, key=abs)
11    print(f'Наименьший по модулю элемент списка: {min_elem}')
12
13    # Находим сумму модулей элементов после первого нуля
14    if 0 in list:
15        idx = list.index(0)
16        sum_elems = sum(abs(x) for x in list[idx + 1:])
17        print(f'Сумма модулей элементов списка, расположенных после первого элемента, равного нулю: {sum_elems}')
18    else:
19        print('В списке нет элементов, равных нулю.')
20
21    # Преобразуем список
22    new_list = list[::2] + list[1::2]
23    print(f'Преобразованный список: {new_list}')
24
25 if __name__ == '__main__':
```

Run: individual2

C:\Users\Gaming-PC\Lab2.4\venv\Scripts\python.exe C:\Users\Gaming-PC\Lab2.4\individual2.py

Введите список целых чисел через пробел: 11 1 5 3 2 0 2 4

Наименьший по модулю элемент списка: 0

Сумма модулей элементов списка, расположенных после первого элемента, равного нулю: 17

Преобразованный список: [11, 1, 5, 3, 2, 0, 2, 4]

Process finished with exit code 0

Рисунок 8. Программа с нулевым элементом

Задание 6.

После выполнения работы на ветке develop, слил ее с веткой main и отправил изменения на удаленный сервер.

```
C:\Users\Gaming-PC\Laba_2.4>git merge develop
Updating 0e721bd..52f184a
Fast-forward
 .idea/.gitignore | 0
 individual1.py   | 15 +++++
 individual2.py   | 24 +++++
 primer1.py       | 18 +++++
 primer2.py       | 32 +++++
 5 files changed, 89 insertions(+)
 create mode 100644 .idea/.gitignore
 create mode 100644 individual1.py
 create mode 100644 individual2.py
 create mode 100644 primer1.py
 create mode 100644 primer2.py
```

Рисунок 9. Слияние ветки develop и main

Ссылка: https://github.com/EvgenyEvdakov/Laba_2.4

Ответы на контрольные вопросы:

1. Что такое списки в языке Python?

Список - это один из встроенных типов данных в языке Python, представляющий собой упорядоченный изменяемый набор объектов произвольных типов. В нем можно хранить объекты различных типов. 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?

Списки в Python хранятся в оперативной памяти в виде последовательности элементов, каждый из которых может быть любого типа. При создании списка выделяется некоторое количество памяти, которое может быть увеличено или уменьшено в зависимости от изменения размера списка.

4. Каким образом можно перебрать все элементы списка?

Для перебора всех элементов списка можно использовать цикл `for` или метод `for in`:

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?

Чтобы проверить, содержится ли элемент в списке, можно использовать оператор `in`.

7. Как определить число вхождений заданного элемента в списке?

Чтобы определить число вхождений заданного элемента в списке, можно использовать метод `count()`

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

Чтобы добавить элемент в список, можно использовать метод `append()`

Чтобы вставить элемент в список на заданную позицию, можно использовать метод `insert()`

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?

Для удаления одного элемента из списка можно использовать метод `remove()`, указав в скобках значение элемента, который нужно удалить. Удалить элемент можно, написав его индекс в методе `pop`. Если не указывать индекс, то функция удалит последний элемент.

11. Что такое списковое включение и как с его помощью осуществлять обработку 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. Как осуществляется доступ к элементам списков с помощью срезов?

Доступ к элементам списка с помощью срезов осуществляется с помощью квадратных скобок []. Срезы задаются в виде start:stop:step, где start - индекс первого элемента в срезе (включительно), stop - индекс последнего элемента в срезе (не включительно), а step - шаг, с которым нужно выбирать элементы из списка.

13. Какие существуют функции агрегации для работы со списками?

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.

14. Как создать копию списка?

copy.copy(x)

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

Вывод: приобрел навыки по работе с кортежами при написании программ с помощью языка программирования Python версии 3.x.