

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное автономное образовательное учреждение  
высшего образования  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии  
Департамент цифровых, робототехнических систем и электроники

**ОТЧЕТ**  
**ПО ПРАКТИЧЕСКОЙ РАБОТЕ №4**  
дисциплины «**Программирование на Python**»  
Вариант №11

Выполнила:

Ковжого Елизавета Андреевна

2 курс, группа ИВТ-б-о-24-1,

09.03.01 «Информатика и вычислительная техника», направленность (профиль)  
«Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

---

(подпись)

Проверил:

Воронкин Р. А., доцент  
департамента цифровых,  
робототехнических систем и  
электроники института  
перспективной инженерии.

---

(подпись)

Отчет защищен с оценкой \_\_\_\_\_ Дата защиты \_\_\_\_\_

Ставрополь, 2025 г.

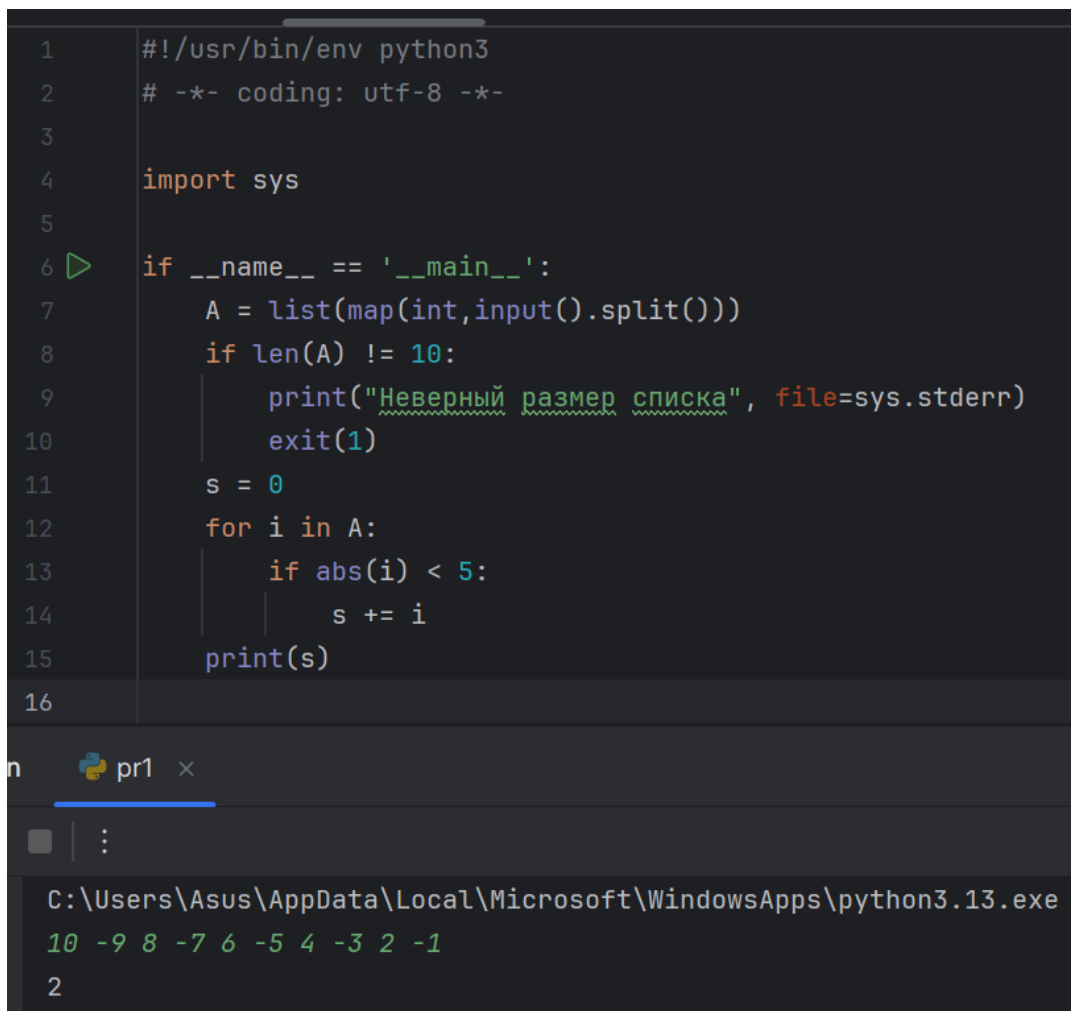
Тема: работа со списками и кортежами в языке Python.

Цель: приобретение навыков по работе со списками и кортежами при написании программ на языке программирования Python версии 3.x.

Порядок выполнения работы:

Адрес репозитория: [https://github.com/LissKovzogo/Python\\_LAB\\_4.git](https://github.com/LissKovzogo/Python_LAB_4.git)

1. Создали, настроили и клонировали репозиторий Python\_LAB\_4.
2. Создали проект PyCharm в папке репозитория.
3. Проработали все примеры лабораторной работы и создали для каждого отдельный модуль Python.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      A = list(map(int, input().split()))
8      if len(A) != 10:
9          print("Неверный размер списка", file=sys.stderr)
10         exit(1)
11         s = 0
12         for i in A:
13             if abs(i) < 5:
14                 s += i
15         print(s)
16
```

The screenshot shows a code editor with a Python script. The script prompts the user to input 10 integers. If the input is not 10 integers, it prints an error message. Otherwise, it calculates the sum of integers whose absolute value is less than 5. The terminal output shows the input sequence: 10 -9 8 -7 6 -5 4 -3 2 -1, and the resulting sum: 2.

Рис. 1 — Работа с примером № 1

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  ▶ if __name__ == '__main__':
7      a = list(map(int, input().split()))
8      if not a:
9          print("Error", file = sys.stderr)
10         exit(1)
11     a_min = a_max = a[0]
12     i_min = i_max = 0
13     for i, item in enumerate(a):
14         if item < a_min:
15             i_min, a_min = i, item
16
17         if item >= a_max:
18             i_max, a_max = i, item
19
20     if i_max < i_min:
21         i_max, i_min = i_min, i_max
22
23     count = 0
24     for item in a[i_min+1:i_max]:
25         if item > 0:
26             count += 1
27
28     print(count)
```

pr2 x


C:\Users\Asus\AppData\Local\Microsoft\WindowsApps\python3.10.0\python3.10.0.exe




-10 7 3 4 15 6 1 1 2 14

3

Рис. 2 — Работа с примером № 2

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5
6  if __name__ == '__main__':
7      A = tuple(map(int, input().split()))
8      if len(A) != 10:
9          print("Неверный размер кортежа", file=sys.stderr)
10         exit(1)
11
12         s = 0
13     for i in A:
14         if abs(i) < 5:
15             s += i
16     print(s)
```

Run  pr3 x

C:\Users\Asus\AppData\Local\Microsoft\WindowsApps\python3.13.6  
10 9 6 8 1 3 7 1 -4 -1  
0

Рис. 3 — Работа с примером № 3

4. Выполнили индивидуальные задания согласно варианту.

Индивидуальное задание №1: Ввести список A из 10 элементов, найти сумму отрицательных элементов кратных 7. их количество и вывести результаты на экран.

Листинг кода задания №1:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == '__main__':
    A = list(map(int, input().split()))
    sum_A = 0
```

```

count_A = 0
for i in A:
    if (i < 0) and (i % 7 == 0):
        count_A += 1
        sum_A += i
print(f"Количество = {count_A}\n Сумма = {sum_A}")

```

```

1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  if __name__ == '__main__':
5
6      A = list(map(int, input().split()))
7      sum_A = 0
8      count_A = 0
9      for i in A:
10         if (i < 0) and (i % 7 == 0):
11             count_A += 1
12             sum_A += i
13
14     print(f"Количество = {count_A}\n Сумма = {sum_A}")

```

Run ind\_1 x

```

C:\Users\Asus\AppData\Local\Microsoft\WindowsApps\python3.
-7 2 -49 49 7
Количество = 2
Сумма = -56

```

Рис. 4 — Индивидуальное задание №1

Индивидуальное задание №2: В списке, состоящем из вещественных элементов, вычислить:

1. номер максимального по модулю элемента списка;
2. сумму элементов списка, расположенных после первого положительного элемента.

Преобразовать список таким образом, чтобы сначала располагались все элементы, целая часть которых лежит в интервале  $[a, b]$ , а потом все остальные.

Листинг кода задания №2:

```

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

```

```
if __name__ == '__main__':  
    a = list(map(int, input("Введите список: ").split()))  
    s = int(input("Ведите нижнюю границу диапазона: "))  
    b = int(input("Ведите верхнюю границу диапазона: "))  
  
    min_a = min(a)  
    max_a = max(a)  
    ind = a.index(max_a)  
    sum_a = 0  
    if abs(min_a) > max_a:  
        ind = a.index(min_a)  
    ind_s = -1  
    for i in a:  
        if i > 0:  
            ind_s = a.index(i)  
            break  
    sum_a = sum(a[ind_s+1::])  
    a = sorted(a, key=lambda x: (0 if s <= x <= b else 1, x))  
    print(ind)  
    print(sum_a)  
    print(a)
```

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  if __name__ == '__main__':
5
6      a = list(map(int, input("Введите список: ").split()))
7      s = int(input("Введите нижнюю границу диапазона: "))
8      b = int(input("Введите верхнюю границу диапазона: "))
9
10     min_a = min(a)
11     max_a = max(a)
12     ind = a.index(max_a)
13     sum_a = 0
14
15     if abs(min_a) > max_a:
16         ind = a.index(min_a)
17
18     ind_s = -1
19
20     for i in a:
21         if i > 0:
22             ind_s = a.index(i)
23             break
24
25     sum_a = sum(a[ind_s+1:])
26     a = sorted(a, key=lambda x: (0 if s <= x <= b else 1, x))
27
28     print("номер максимального по модулю элемента списка", ind)
29     print("сумму элементов списка, расположенных после первого положительного элемента", sum_a)
30     print(a)
```

Run ind\_2 x

Введите список: 4 2 9 4 1 -300 38 18 4  
Введите нижнюю границу диапазона: 5  
Введите верхнюю границу диапазона: 20  
номер максимального по модулю элемента списка 5  
сумму элементов списка, расположенных после первого положительного элемента -224  
[-300, 38, 18, 4]

Рис. 5 — Индивидуальное задание №2

Индивидуальное задание №3: Имеются данные о сумме очков, набранных в чемпионате каждой из футбольных команд. Определить, перечислены ли команды в списке в соответствии с занятыми ими местами в чемпионате.

Листинг кода задания №3:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
if __name__ == '__main__':
    teams = (
```

```

("Команда А", 15),
("Команда Б", 12),
("Команда В", 18),
("Команда Г", 10),
("Команда Д", 14)
)

sorted_teams = sorted(teams, key=lambda x: x[1], reverse=True)
print(sorted_teams)

for i, (team, point) in enumerate(sorted_teams):
    # print(f"Место: {i+1} {team} {point}")
    print(f"Место: {i + 1} {sorted_teams[i][0]} {sorted_teams[i][1]}")
    # print(f"Место: {i + 1} {sorted_teams[i]} ")

```

```

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

if __name__ == '__main__':
    teams = (
        ("Команда А", 15),
        ("Команда Б", 12),
        ("Команда В", 18),
        ("Команда Г", 10),
        ("Команда Д", 14)
    )

    sorted_teams = sorted(teams, key=lambda x: x[1], reverse=True)
    print(sorted_teams)
    for i, (team, point) in enumerate(sorted_teams):
        # print(f"Место: {i+1} {team} {point}")
        print(f"Место: {i + 1} {sorted_teams[i][0]} {sorted_teams[i][1]}")

```

Output:

```

[('Команда В', 18), ('Команда А', 15), ('Команда Д', 14), ('Команда Б', 12), ('Команда Г', 10)]
Место: 1 Команда В 18
Место: 2 Команда А 15
Место: 3 Команда Д 14
Место: 4 Команда Б 12
Место: 5 Команда Г 10

```

Рис. 6 — Индивидуальное задание №3

5. Зафиксировали все изменения в репозиторий.



```

Asus@DESKTOP-KEJGOCU MINGW64 ~/gitprogect/Python_LAB_4 (main)
$ git commit -m"подготовка отчёта"
[main da7397e] подготовка отчёта
24 files changed, 108 insertions(+), 60 deletions(-)
rename {PyCharm/.idea => .idea}/.gitignore (100%)
rename PyCharm/.idea/PyCharm.iml => .idea/Python_LAB_4.iml (100%)
rename {PyCharm/.idea => .idea}/inspectionProfiles/profiles_settings.xml (100%)
rename {PyCharm/.idea => .idea}/misc.xml (100%)
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 Indiv/.idea/.gitignore
create mode 100644 Indiv/.idea/PyCharm.iml
create mode 100644 Indiv/.idea/inspectionProfiles/profiles_settings.xml
create mode 100644 Indiv/.idea/misc.xml
rename {PyCharm => Indiv}/.idea/modules.xml (100%)
rename {PyCharm => Indiv}/.idea/vcs.xml (100%)
create mode 100644 Indiv/ind_1.py
create mode 100644 Indiv/ind_2.py
create mode 100644 Indiv/ind_3.py
rename {PyCharm => Prim}/pr1.py (85%)
rename {PyCharm => Prim}/pr2.py (91%)
rename {PyCharm => Prim}/pr3.py (85%)
delete mode 100644 PyCharm/ind_1.py
delete mode 100644 PyCharm/ind_2.py
delete mode 100644 PyCharm/ind_3.py
delete mode 100644 PyCharm/main.py
create mode 100644 "doc/Python_\320\277\321\200\320\276\320\263\321\200\320\260\320\274\320\274\320\27
0\321\200\320\276\320\262\320\260\320\275\320\270\320\265_4.doc"
create mode 100644 "doc/~$thon_\320\277\321\200\320\276\320\263\321\200\320\260\320\274\320\274\320\27
0\321\200\320\276\320\262\320\260\320\275\320\270\320\265_4.doc"

```

Рис. 16 — Фиксирование изменений

### Контрольные вопросы:

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

Список — это упорядоченная коллекция элементов, которая может хранить данные разных типов. Списки можно изменять после создания.

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

Список создается с помощью квадратных скобок, например: `my_list = [1, "текст", 3.14]`. Также можно использовать функцию `list()`.

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

Списки хранятся как динамические массивы ссылок на объекты. Это позволяет эффективно добавлять новые элементы и изменять существующие.

#### 4. Каким образом можно перебрать все элементы списка?

Для перебора используется цикл `for`: `for element in my_list: print(element)`

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

Списки можно складывать (объединять) и умножать на число (повторять элементы).

#### 6. Как проверить есть ли элемент в списке?

С помощью оператора ``in``: ``if element in my_list:``

7. Как определить число вхождений заданного элемента в списке?

Методом ``count()``: ``my_list.count(5)`` покажет сколько раз 5 встречается в списке.

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

``append()`` добавляет в конец, ``insert()`` вставляет на конкретную позицию, ``extend()`` добавляет несколько элементов.

9. Как выполнить сортировку списка?

Метод ``sort()`` сортирует исходный список, а функция ``sorted()`` возвращает новый отсортированный список.

10. Как удалить один или несколько элементов из списка?

``remove()`` удаляет первый найденный элемент, ``pop()`` удаляет по индексу, ``del`` удаляет по индексу или срезу.

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

Это компактный способ создания списков: ``[x*2 for x in range(5)]`` создаст список ``[0, 2, 4, 6, 8]``.

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

Через указание начала, конца и шага: ``my_list[начало:конец:шаг]``. Например, ``my_list[1:5:2]``.

13. Какие существуют функции агрегации для работы со списками?

``len()`` — длина списка, ``sum()`` — сумма чисел, ``min()`` и ``max()`` — минимальный и максимальный элемент.

14. Как создать копию списка?

Через метод ``copy()`` или срез ``my_list[:]``. Для вложенных списков нужна ``deepcopy()`` из модуля `copy`.

15. В чем отличие функции `sorted` от метода `sort` списков?

``sort()`` изменяет исходный список, а ``sorted()`` возвращает новый отсортированный список, не меняя исходный.

16. Что такое кортежи в языке Python?

Кортеж — это упорядоченная коллекция элементов, которая не может быть изменена после создания.

17. Каково назначение кортежей в языке Python?

Кортежи используются для хранения данных, которые не должны изменяться, и могут быть ключами словаря.

18. Как осуществляется создание кортежей?

Через круглые скобки: ``my_tuple = (1, 2, 3)`` или без скобок: ``my_tuple = 1, 2, 3``.

19. Как осуществляется доступ к элементам кортежа?

Так же как к элементам списка — по индексу: ``my_tuple[0]`` вернет первый элемент.

20. Зачем нужна распаковка (деструктуризация) кортежа?

Чтобы присвоить элементы кортежа отдельным переменным: ``a, b, c = (1, 2, 3)``.

21. Какую роль играют кортежи в множественном присваивании?

Множественное присваивание ``x, y = 10, 20`` фактически создает кортеж ``(10, 20)`` и распаковывает его.

22. Как выбрать элементы кортежа с помощью среза?

Так же как у списков: ``my_tuple[1:3]`` вернет элементы с 1 по 2 индекс.

23. Как выполняется конкатенация и повторение кортежей?

С помощью операторов ``+`` (сложение) и ``*`` (умножение на число), создаются новые кортежи.

24. Как выполняется обход элементов кортежа?

Циклом `for``: ``for item in my_tuple: print(item)``

25. Как проверить принадлежность элемента кортежу?

Оператором ``in``: ``if 5 in my_tuple:``

26. Какие методы работы с кортежами Вам известны?

Только `'count()'` для подсчета вхождений и `'index()'` для поиска индекса элемента.

27. Допустимо ли использование функций агрегации таких как `len()`, `sum()` и т.п. при работе с кортежами?

Да, все эти функции работают с кортежами так же как со списками.

28. Как создать кортеж с помощью спискового включения?

Нужно использовать генератор в функции `tuple()`: `'tuple(x**2 for x in range(5))'`

Вывод: приобрели навыков по работе со списками и кортежами при написании программ на языке программирования Python версии 3.x.