

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ  
ФЕДЕРАЦИИ**

**Федеральное государственное автономное  
образовательное учреждение высшего  
образования  
«СЕВЕРОКАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ  
УНИВЕРСИТЕТ»**

**Кафедра**

**инфокоммуникаций**

**Институт цифрового**

**развития**

**ОТЧЁТ**

**по лабораторной работе №2.4**

Дисциплина: «Основы программной инженерии»

Тема: «Работа со списками в языке Python»

Выполнил:

студент 2 курса

группы Пиж-б-о-21-1

Коныжев Максим

Викторович

Ставрополь 2022

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


Ход работы:

1. Был создан репозиторий в Github в который были добавлены правила gitignore для работы IDE PyCharm, была выбрана лицензия MIT, сам репозиторий был клонирован на локальный сервер и был организован в соответствии с моделью ветвления git-flow.

Import a repository.

---


Owner \*      Repository name \*


 MaxDrill ▾ / lab2\_4 ✓

Great repository names are short and memorable. Need inspiration? How about [cautious-rotary-phone?](#)

Description (optional)

---

☒  **Public**  
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**  
You choose who can see and commit to this repository.

---

**Initialize this repository with:**  
Skip this step if you're importing an existing repository.


☒ **Add a README file**  
This is where you can write a long description for your project. [Learn more.](#)

**Add .gitignore**  
Choose which files not to track from a list of templates. [Learn more.](#)


**.gitignore template:** Python ▾

**Choose a license**  
A license tells others what they can and can't do with your code. [Learn more.](#)

**License:** MIT License ▾

This will set  **main** as the default branch. Change the default name in your [settings](#).

---

 You are creating a public repository in your personal account.

---

**Create repository**

Рисунок 1.1 – Создание репозитория

```
C:\Users\UESR\gitproj>git clone https://github.com/MaxDrill/lab2_4.git
Cloning into 'lab2_4'...
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 5 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (5/5), done.

C:\Users\UESR\gitproj>
```

Рисунок 1.2 – Клонирование репозитория

```
C:\Users\UESR\gitproj\lab2_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/UESR/gitproj/lab2_4/.git/hooks]

C:\Users\UESR\gitproj\lab2_4>
```

Рисунок 1.3 – Организация репозитория в соответствии с моделью ветвления  
git-flow

2. Был создана папка PyCharm в которой хранятся примеры из лабораторной работы.

```
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         # Найти искомую сумму.
13         s = sum([a for a in A if abs(a) < 5])
14         print(s)
15
16     if __name__ == '__main__'
```

num1 x

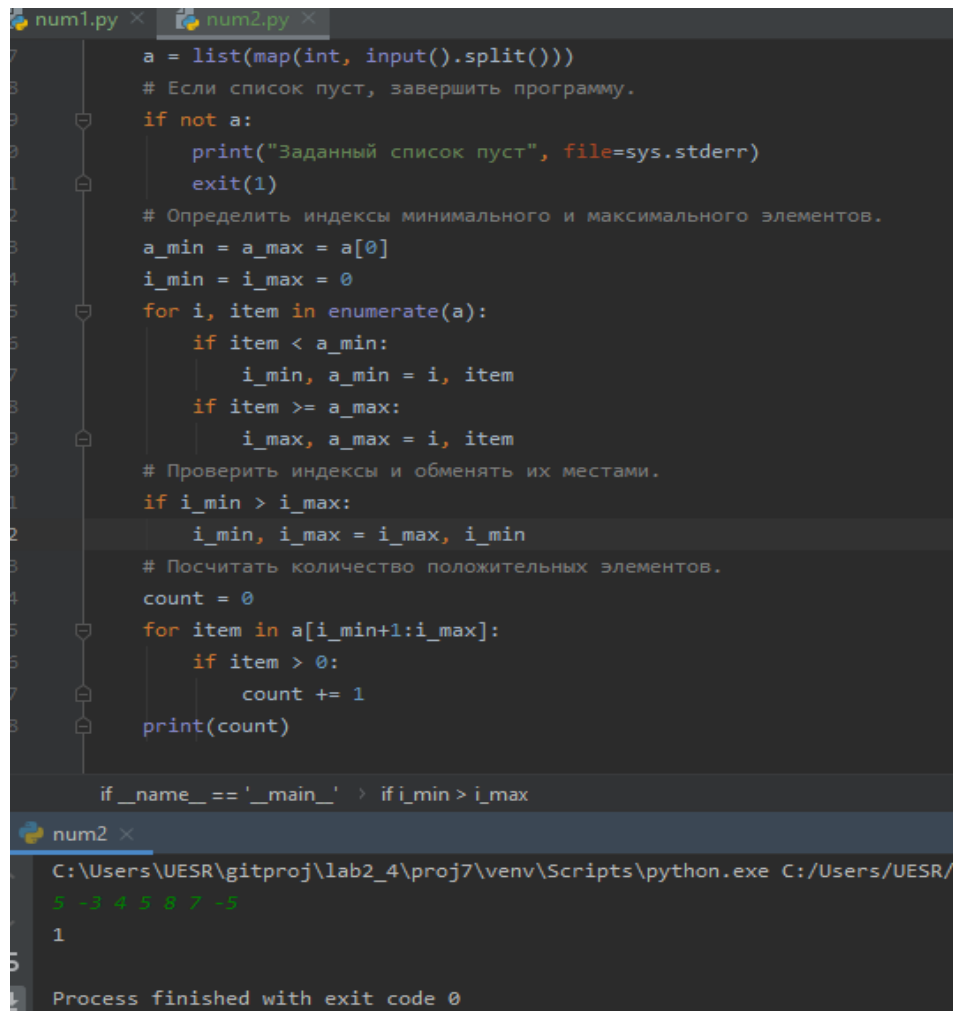
C:\Users\UESR\gitproj\lab2\_4\proj7\venv\Scripts\python.exe C:/Users/UE

1 3 2 4 5 6 7 8 9 5

10

Process finished with exit code 0

Рисунок 2.1 – Результат работы первого примера



```
num1.py x num2.py x
7 a = list(map(int, input().split()))
8 # Если список пуст, завершить программу.
9 if not a:
10     print("Заданный список пуст", file=sys.stderr)
11     exit(1)
12 # Определить индексы минимального и максимального элементов.
13 a_min = a_max = a[0]
14 i_min = i_max = 0
15 for i, item in enumerate(a):
16     if item < a_min:
17         i_min, a_min = i, item
18     if item >= a_max:
19         i_max, a_max = i, item
20 # Проверить индексы и обменять их местами.
21 if i_min > i_max:
22     i_min, i_max = i_max, i_min
23 # Посчитать количество положительных элементов.
24 count = 0
25 for item in a[i_min+1:i_max]:
26     if item > 0:
27         count += 1
28 print(count)

if __name__ == '__main__': > if i_min > i_max

num2 x
C:\Users\UESR\gitproj\lab2_4\proj7\venv\Scripts\python.exe C:/Users/UESR/
5 -3 4 5 8 7 -5
1
Process finished with exit code 0
```

Рисунок 2.2 – Результат работы второго примера

3. Было выполнено два индивидуальных задания согласно 7 варианту

#### Задание 1

Ввести список А из 10 элементов, найти произведение отрицательных элементов и вывести его на экран.

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

if __name__ == '__main__':
    A = list(map(int, input().split()))
```

```

if len(A) != 10:
    print("there are't 10 items in the list", file=sys.stderr)
    exit(1)
p = 1
for i in A:
    if i < 0:
        p *= i
print(p)

```

```

1 2 3 4 5 6 7 8 -2 -3
6
Process finished with exit code 0

```

Рисунок 3.1 – Результат работы программы

## Задание 2

В списке, состоящем из вещественных элементов, вычислить:

1. номер минимального элемента списка;
2. сумму элементов списка, расположенных между первым и вторым отрицательными элементами.

Преобразовать список таким образом, чтобы сначала располагались все элементы, модуль которых не превышает 1, а потом - все остальные.

```

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

if __name__ == '__main__':

```

```

a = list(map(float, input("Input list: ").split()))
a1 = []
min_a = a[0]
min_i = 0
s = 0

for i in range(0, len(a)):
    if a[i] < min_a:
        min_a = a[i]
        min_i = i

for i in range(0, len(a)):
    if a[i] < 0:
        a1.append(i)

w1 = a1[0]
w2 = a1[1]

for i in a[a1[0] + 1: a1[1]]:
    s += i

a.sort(key=abs)

print(' '.join(map(str, a)))
print("Index of min elem:", min_i, "\n", "Sum elem:", s)

```

```

Input list: -5 5 6 7 0.5 -0.1 -9
-0.1 0.5 -5.0 5.0 6.0 7.0 -9.0
Index of min elem: 6
Sum elem: 18.5

```

Рисунок 3.2 – Результат работы программы

4. Был осуществлен коммит и слияние веток main и develop, также были запущены изменения на удаленный сервер.

```

C:\Users\UESR\gitproj\lab2_4>git commit -m "Add proj"
[develop e11456d] Add proj
4 files changed, 86 insertions(+)
create mode 100644 proj7/idz1.py
create mode 100644 proj7/idz2.py
create mode 100644 proj7/num1.py
create mode 100644 proj7/num2.py

C:\Users\UESR\gitproj\lab2_4>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

```

Рисунок 4.1 – Коммит изменений и переход на ветку main

```
C:\Users\UESR\gitproj\lab2_4>git merge develop
Updating 074737d..e11456d
Fast-forward
 proj7/idz1.py | 14 ++++++++
 proj7/idz2.py | 30 ++++++++
 proj7/num1.py | 14 ++++++++
 proj7/num2.py | 28 ++++++++
4 files changed, 86 insertions(+)
create mode 100644 proj7/idz1.py
create mode 100644 proj7/idz2.py
create mode 100644 proj7/num1.py
create mode 100644 proj7/num2.py

C:\Users\UESR\gitproj\lab2_4>git branch
  develop
* main

C:\Users\UESR\gitproj\lab2_4>git push
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (7/7), 1.71 KiB | 583.00 KiB/s, done.
Total 7 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/MaxDrill/lab2_4.git
   074737d..e11456d  main -> main

C:\Users\UESR\gitproj\lab2_4>
```

Рисунок 4.2 – Слияние веток и пуш изменений на удаленный сервер

Вывод: были приобретены навыки по работе со списками при написании программ с помощью языка программирования Python версии 3.x.



## Контрольные вопросы

1. Что такое списки в языке Python?

Список (list) – это структура данных для хранения объектов различных типов.

2. Как осуществляется создание списка в Python?

Для создания списка нужно заключить элементы в квадратные скобки.

3. Как организовано хранение списков в оперативной памяти?

Список является изменяемым типом данных. При его создании 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?

```
for elem in my_list:
```

5. Какие существуют арифметические операции со списками?

+, \*

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?

```
list.count('элемент')
```

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

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

9. Как выполнить сортировку списка?

`list.sort()`

10. Как удалить один или несколько элементов из списка?

Удалить элемент можно, написав его индекс в методе `pop`.

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.

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

`list[<начало среза>:<конец среза>:<шаг>]`

13. Какие существуют функции агрегации для работы со списками?

Для работы со списками Python предоставляет следующие функции:

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

содержит только числовые значения

14. Как создать копию списка?

Для создания копии списка необходимо использовать либо метод `copy`, либо использовать оператор среза

15. Самостоятельно изучите функцию `sorted` языка Python. В чем ее

отличие от метода `sort` списков?

Отличие заключается в том, что метод `list.sort()` определён только для списков, в то время как `sorted()` работает со всеми итерируемыми объектами.