

**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»

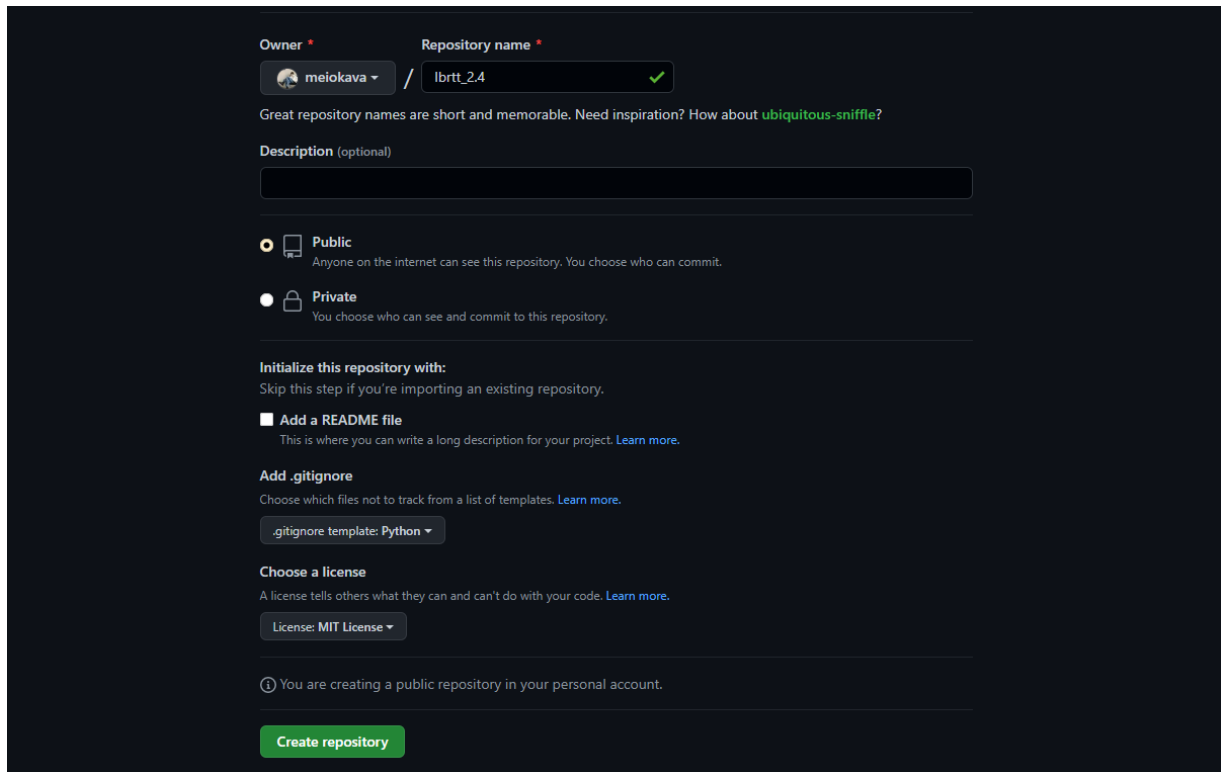
Выполнила:
студентка 2 курса
группы Пиж-б-о-21-1
Джолдошова Мээрим
Бекболотовна

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.





Owner ^{*} Repository name ^{*}

 meiokava / lbrtt_2.4 

Great repository names are short and memorable. Need inspiration? How about [ubiquitous-sniffle](#)?

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 

 You are creating a public repository in your personal account.

[Create repository](#)

Рисунок 1.1 – Создание репозитория

```
C:\Users\мвидео>cd/d C:\lbrtt_2.4
C:\lbrtt_2.4>git clone https://github.com/meiokava/lbrtt_2.4.git
Cloning into 'lbrtt_2.4'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (4/4), done.
```

Рисунок 1.2 – Клонирование репозитория

```

C:\lbrtt_2.4\lbrtt_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:/lbrtt_2.4/lbrtt_2.4/.git/hooks]

```

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

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

```

C:\Users\мвидео\PycharmProjects\pythonProject35\venv\Scripts\python.exe C:\Users\мвидео\PycharmProjects\pythonProject35\venv\Scripts\python.exe
1 3 4 5 6 7 8 9 3 4
15
Process finished with exit code 0

```

Рисунок 2.1 – Результат работы первого примера

```

C:\Users\мвидео\PycharmProjects\pythonProject36\venv\Scripts\python.exe C:\Users\мвидео\PycharmProjects\pythonProject36\venv\Scripts\python.exe
5 -3 4 5 8 7 -5
1
Process finished with exit code 0

```

Рисунок 2.2 – Результат работы второго примера



« Windows (C:) » lbrtt_2.4 » lbrtt_2.4 » PyCharm	
Имя	Дата изменения
 exmpl1.py	12.11.2022 10:07
 exmpl2.py	12.11.2022 10:18

Рисунок 2.3 – Папка с примерами из лабораторной работы

3. Было выполнено два индивидуальных задания согласно 5 варианту

Задание 1

Ввести список A из 10 элементов, найти сумму элементов, больших 3 и меньших 8 и вывести ее 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("size isn't right", file=sys.stderr)
        exit(1)
    s = 0
    for item in A:
        if (abs(item) > 3) and (abs(item) < 8):
            s += item
    print(s)
```

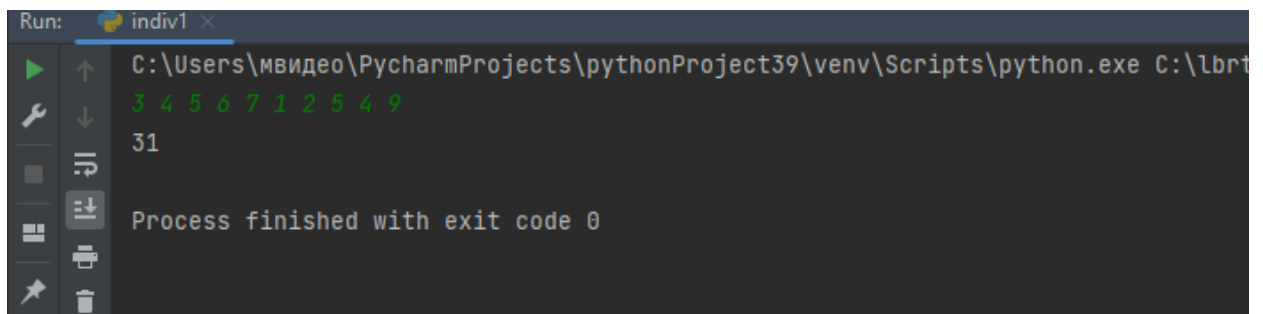


Рисунок 3.1 – Результат работы программы

Задание 2

В списке, состоящем из вещественных элементов, вычислить:

1. максимальный элемент списка;
2. сумму элементов списка, расположенных до последнего положительного элемента.

Сжать список, удалив из него все элементы, модуль которых находится в интервале $[a, b]$.

Освободившиеся в конце списка элементы заполнить нулями.

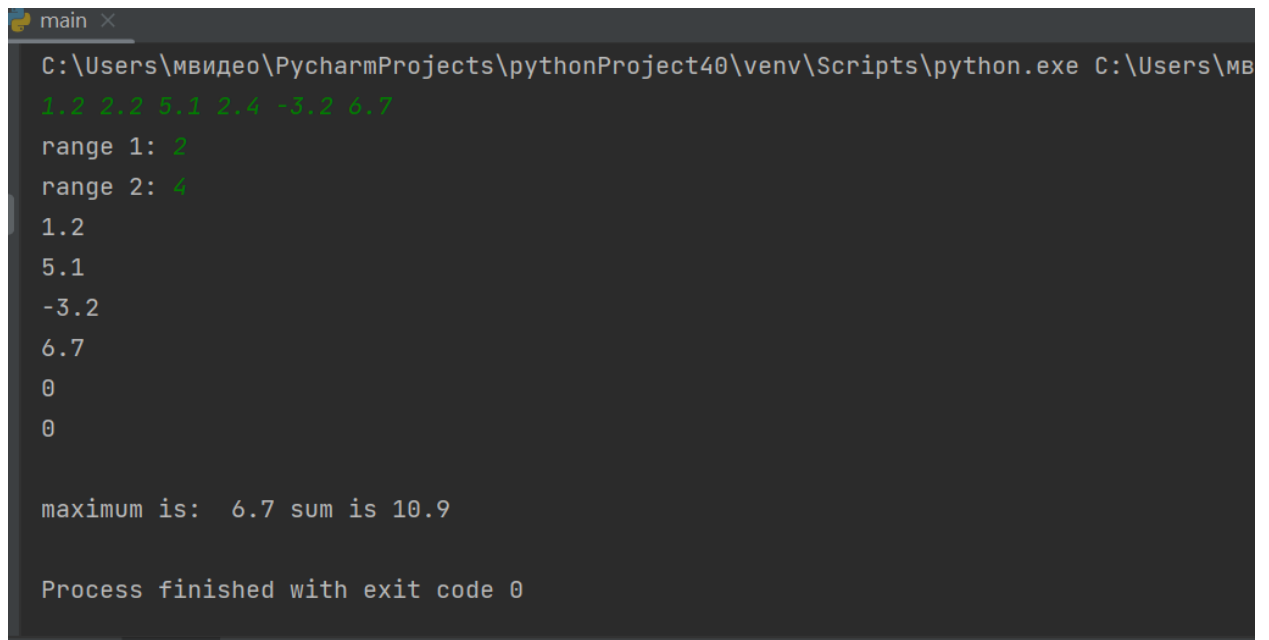
```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import sys

if __name__ == '__main__':
```

```

a = list(map(float, input().split()))
a1 = list(range(0, len(a)))
in1 = int(input("range 1: "))
in2 = int(input("range 2: "))
max_ = a[0]
pos = 0
for i in a:
    if i > max_:
        max_ = i
for i in a:
    if i > 0:
        pos += i
    else:
        break
for i in range(0, len(a)):
    if (a[i] >= in1) and (a[i] <= in2):
        a.remove(a[i])
        a.append(0)
for i in range(0, len(a)):
    print(a[i])
print('\nmaximum is: ', max_, 'sum is', pos)

```



```

main x
C:\Users\мvideo\PycharmProjects\pythonProject40\venv\Scripts\python.exe C:\Users\мvideo\PycharmProjects\pythonProject40\main.py
1.2 2.2 5.1 2.4 -3.2 6.7
range 1: 2
range 2: 4
1.2
5.1
-3.2
6.7
0
0
maximum is: 6.7 sum is 10.9
Process finished with exit code 0

```

Рисунок 3.2 – Результат работы программы

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

```

C:\lbrtt_2.4\lbrtt_2.4>git add .
C:\lbrtt_2.4\lbrtt_2.4>git commit -m "new folders"
[develop 5f00bae] new folders
4 files changed, 85 insertions(+)
create mode 100644 Indiv/indiv1.py
create mode 100644 Indiv/indiv2.py
create mode 100644 PyCharm/exmpl1.py
create mode 100644 PyCharm/exmpl2.py
C:\lbrtt_2.4\lbrtt_2.4>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:\lbrtt_2.4\lbrtt_2.4>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\lbrtt_2.4\lbrtt_2.4>git merge develop
Updating 5d580c9..5f00bae
Fast-forward
   Individiv1.py | 14 ++++++++
   Individiv2.py | 26 ++++++++
   Pycharm/exmpl1.py | 17 ++++++++
   Pycharm/exmpl2.py | 28 ++++++++
  4 files changed, 85 insertions(+)
  create mode 100644 Individiv1.py
  create mode 100644 Individiv2.py
  create mode 100644 Pycharm/exmpl1.py
  create mode 100644 Pycharm/exmpl2.py

C:\lbrtt_2.4\lbrtt_2.4>git push
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 1.81 KiB | 371.00 KiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/meiokava/lbrtt_2.4.git
  5d580c9..5f00bae  main -> main
```

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.