

**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 кроссплатформенного программирования»

Тема: «Работа со списками в языке Python»

Выполнил: студент 1 курса

группы ИВТ-б-о-21-1

Криворот Владимир  
Геннадьевич

Ставрополь 2022

## Выполнение работы:

1. Создал репозиторий в GitHub «rep 2.4» в который добавил .gitignore, который дополнил правила для работы с IDE PyCharm с ЯП Python, выбрал лицензию MIT, клонировал его на лок. сервер и организовал в соответствии с моделью ветвления git-flow.

Owner <sup>\*</sup> Repository name <sup>\*</sup>

LenkaGolovach / Laba\_2.4 ✓

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

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](#).

<sup>i</sup> You are creating a public repository in your personal account.

**Create repository**

Рисунок 1.1 Создание репозитория

```
Cloning into 'rep_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

```
D:\git\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? [D:/git/Laba_2.4/.git/hooks]

D:\git\Laba_2.4>
```

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

```
venv/
ENV/
env.bak/
venv.bak/

# Spyder project settings
.spyderproject
.spyproject

# Rope project settings
.ropeproject

# mkdocs documentation
/site

# мурр
.mypy_cache/
.dmypy.json
dmypy.json

# Pyre type checker
.pyre/

# pytype static type analyzer
.pytype/

# Cython debug symbols
cython_debug/

# PyCharm
# JetBrains specific template is maintained in a separate JetBrains.gitignore that can
# be found at https://github.com/github/gitignore/blob/main/Global/JetBrains.gitignore
# and can be added to the global gitignore or merged into this file. For a more nuclear
# option (not recommended) you can uncomment the following to ignore the entire idea folder.
.idea/
```

Рисунок 1.4 Изменение .gitignore

2. Создал проект PyCharm в папке репозитория, проработал примеры  
ЛР.

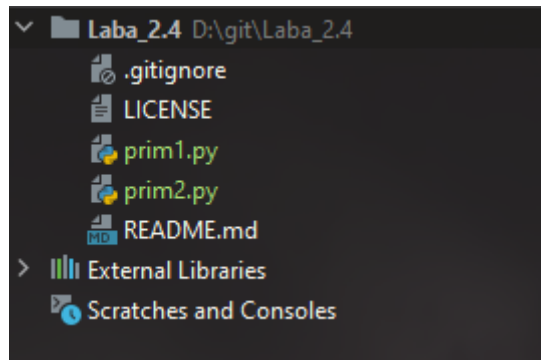


Рисунок 2.1 Создание проекта в PyCharm

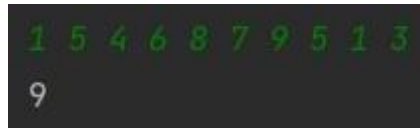


Рисунок 2.2 Рез-т выполнения программы



Рисунок 2.3 Рез-т выполнения программы

3. (16 вариант). Выполнил 2 индивидуальных задания.



Рисунок 3.1 Вывод программы индивидуального задания 1

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

import sys

if __name__ == '__main__':
    A = list(map(int, input().split()))
    if not A:
        print("Заданный список пуст", file=sys.stderr)
        exit(1)

    pol = 0

    for i in range(len(A)):
        if A[i] > 0:
            pol += A[i]

    sum = 0

    for i in range(len(A) - 1, 0, -1):
        if A[i] != 0:
            sum += A[i]
        else:
            break

    print(pol, sum)
```

```
1 -1 2 -2 3 -3 4 -4 0 1 2 3 4
20 10
```

Рисунок 3.2 Вывод программы индивидуального задания 2

4. Сделал коммит, выполнил слияние с веткой main, и запустил изменения в уд. репозиторий.

```
[develop 3356fc8] added programs
1 file changed, 150 insertions(+), 3 deletions(-)

C:\Users\adamk\OneDrive\Рабочий стол\rep_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

```
Updating b61f559..3356fc8
Fast-forward
 .gitignore | 153 ++++++-----
 1 file changed, 150 insertions(+), 3 deletions(-)
```

Рисунок 4.2 Слияние ветки main с develop

```
D:\git\Laba_2.4>git push
Enumerating objects: 15, done.
Counting objects: 100% (15/15), done.
Delta compression using up to 8 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (13/13), 4.84 KiB | 4.84 MiB/s, done.
Total 13 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), done.
To https://github.com/LenkaGolovach/Laba_2.4.git
 1c5f9d6..eb73e21  main -> main
```

Рисунок 4.3 Пуш изменений на удаленный сервер






 <b>LenkaGolovach</b> отсортированные		eb73e21 41 seconds ago	 4 commits
индивидуальные	отсортированные	42 seconds ago	
примеры	отсортированные	42 seconds ago	
 .gitignore	ignore	29 minutes ago	
 LICENSE	Initial commit	31 minutes ago	
 README.md	Initial commit	31 minutes ago	

Рисунок 4.4 Изменения на удаленном сервере

## Контр. вопросы и ответы на них:

### 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.