

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 №7
дисциплины «Основы программной инженерии»

Выполнил:
Матвеев Александр Иванович
1 курс, группа ПИЖ-б-о-22-1,
09.03.04 «Программная инженерия»,
направленность (профиль) «Разработка и
сопровождение программного
обеспечения», очная форма обучения

(подпись)

Проверил Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: Работа со списками в языке Python

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

Ход работы.

1. Создание нового репозитория с лицензией MIT.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Owner * SashkaHacker / **Repository name *** laba3
✔ laba3 is available.

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

Description (optional)
Выполнение лабораторной работы №3 по дисциплине: основы программной инженерии

☒ **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:

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

Add .gitignore
.gitignore template: Python
Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

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

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

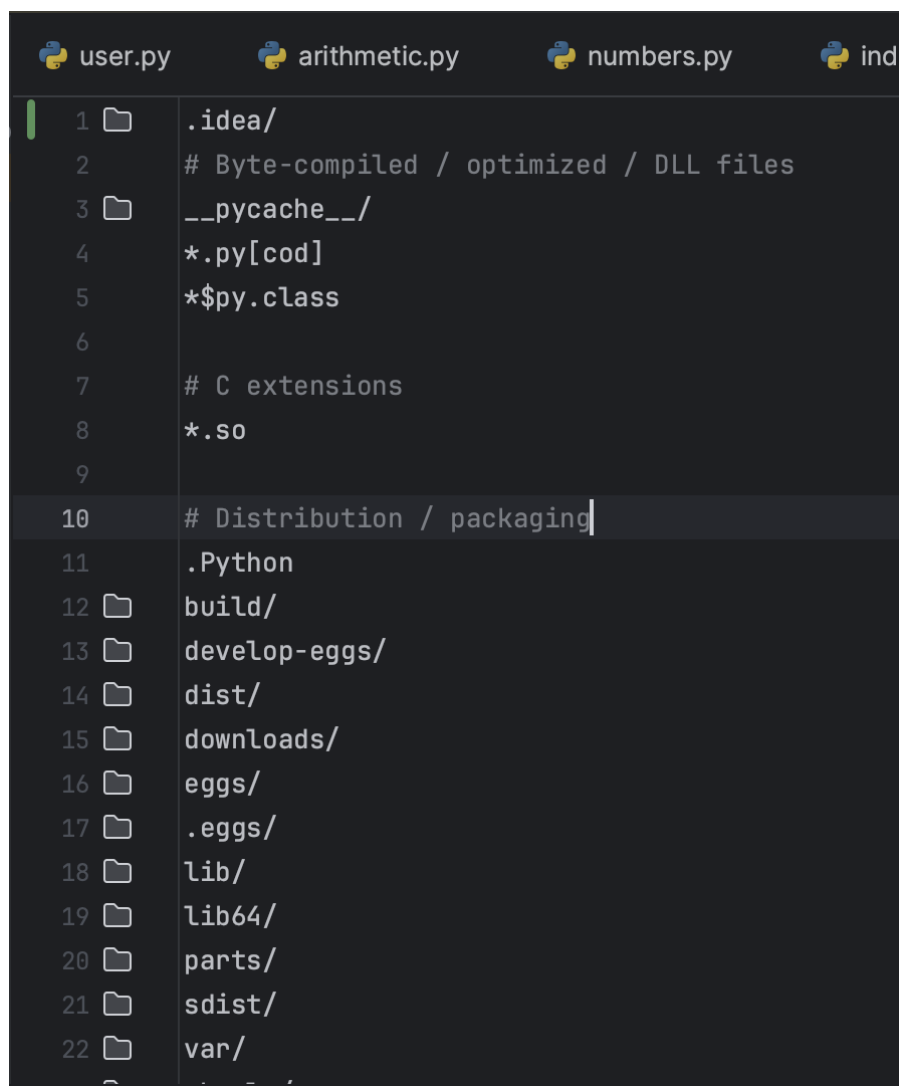
Рисунок 1 – Создание репозитория

2. Клонировал репозиторий на рабочий ПК.

```
Sashka@DESKTOP-U4RPSBI MINGW64 ~/Documents/GitHub/laba6 (develop)
$ git push origin develop
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'develop' on GitHub by visiting:
remote:   https://github.com/SashkaHacker/laba6/pull/new/develop
remote:
To https://github.com/SashkaHacker/laba6.git
 * [new branch]      develop -> develop
```

Рисунок 2 – Клонирование репозитория

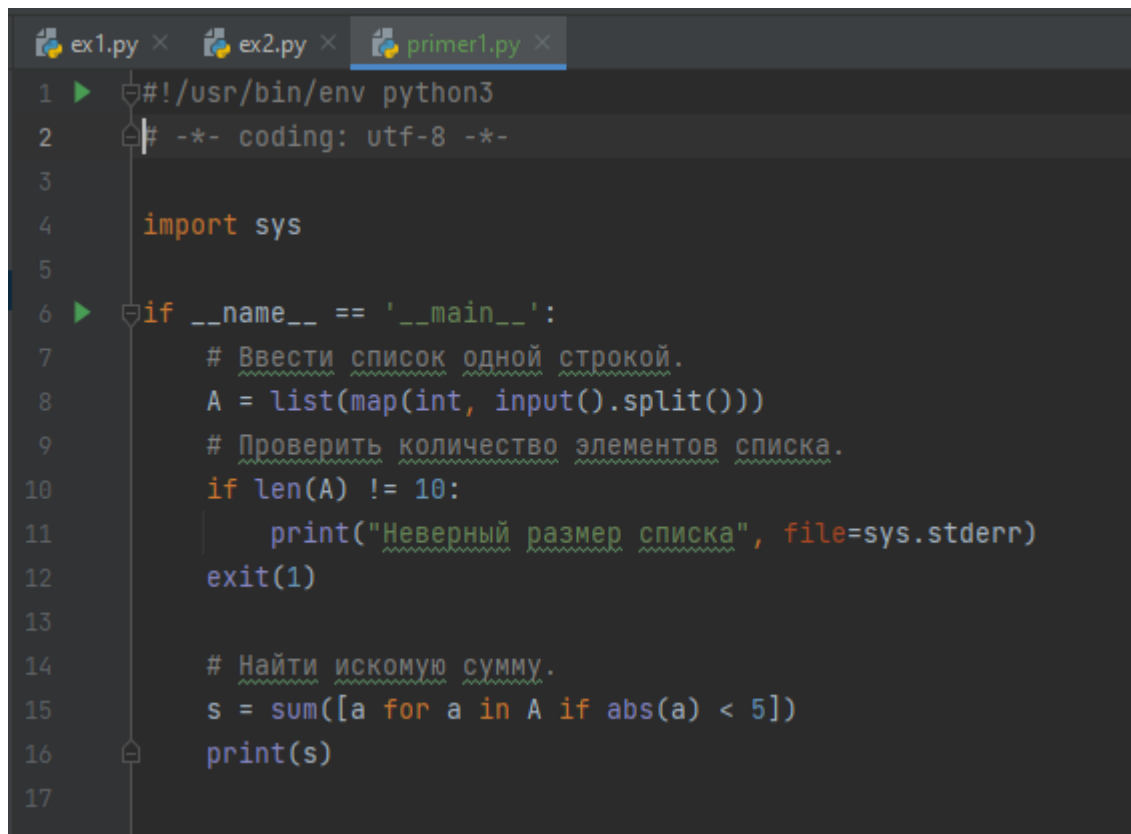
3. Дополнил файл .gitignore необходимыми инструкциями.



```
1  .idea/
2  # Byte-compiled / optimized / DLL files
3  __pycache__/
4  *.py[cod]
5  *$py.class
6
7  # C extensions
8  *.so
9
10 # Distribution / packaging
11 .Python
12 build/
13 develop-eggs/
14 dist/
15 downloads/
16 eggs/
17 .eggs/
18 lib/
19 lib64/
20 parts/
21 sdist/
22 var/
```

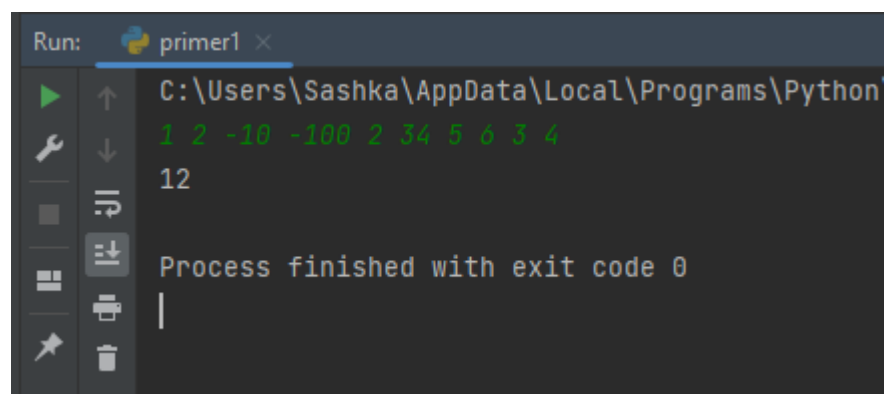
Рисунок 3 – Файл .gitignore

4. Проработка задания №1 из методических указаний.



```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import sys
5
6 if __name__ == '__main__':
7     # Ввести список одной строкой.
8     A = list(map(int, input().split()))
9     # Проверить количество элементов списка.
10    if len(A) != 10:
11        print("Неверный размер списка", file=sys.stderr)
12        exit(1)
13
14    # Найти искомую сумму.
15    s = sum([a for a in A if abs(a) < 5])
16    print(s)
17
```

Рисунок 4 – Код программы



```
Run: primer1 x
C:\Users\Sashka\AppData\Local\Programs\Python
1 2 -10 -100 2 34 5 6 3 4
12
Process finished with exit code 0
|
```

Рисунок 5 – Пример работы программы

5. Проработка задания №2 из методических указаний.

```
ex1.py x ex2.py x primer1.py x primer2.py x
1 ▶ #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 import sys
5
6 ▶ if __name__ == '__main__':
7     # Ввести список одной строкой.
8     a = list(map(int, input().split()))
9     # Если список пуст, завершить программу.
10    if not a:
11        print("Заданный список пуст", file=sys.stderr)
12        exit(1)
13
14    # Определить индексы минимального и максимального элементов.
15    a_min = a_max = a[0]
16    i_min = i_max = 0
17    for i, item in enumerate(a):
18        if item < a_min:
19            i_min, a_min = i, item
20        if item >= a_max:
21            i_max, a_max = i, item
22    # Проверить индексы и обменять их местами.
23    if i_min > i_max:
24        i_min, i_max = i_max, i_min
25    # Посчитать количество положительных элементов.
26    count = 0
27    for item in a[i_min + 1:i_max]:
28        if item > 0:
29            count += 1
30    print(count)
```

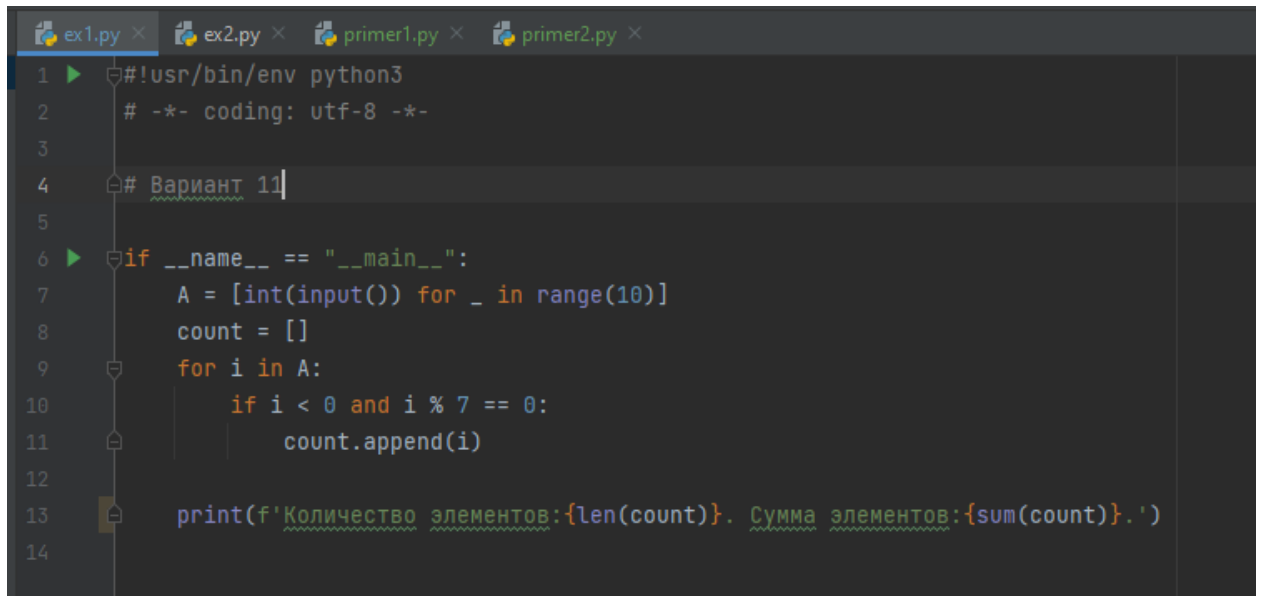
Рисунок 6 – Код программы

```
Run: primer2 x
C:\Users\Sashka\AppData\Local\Programs\Python\Python31
1 -1000 4 100000 32 4 5 34 100000000000
6
Process finished with exit code 0
```

Рисунок 7 – Пример работы программы

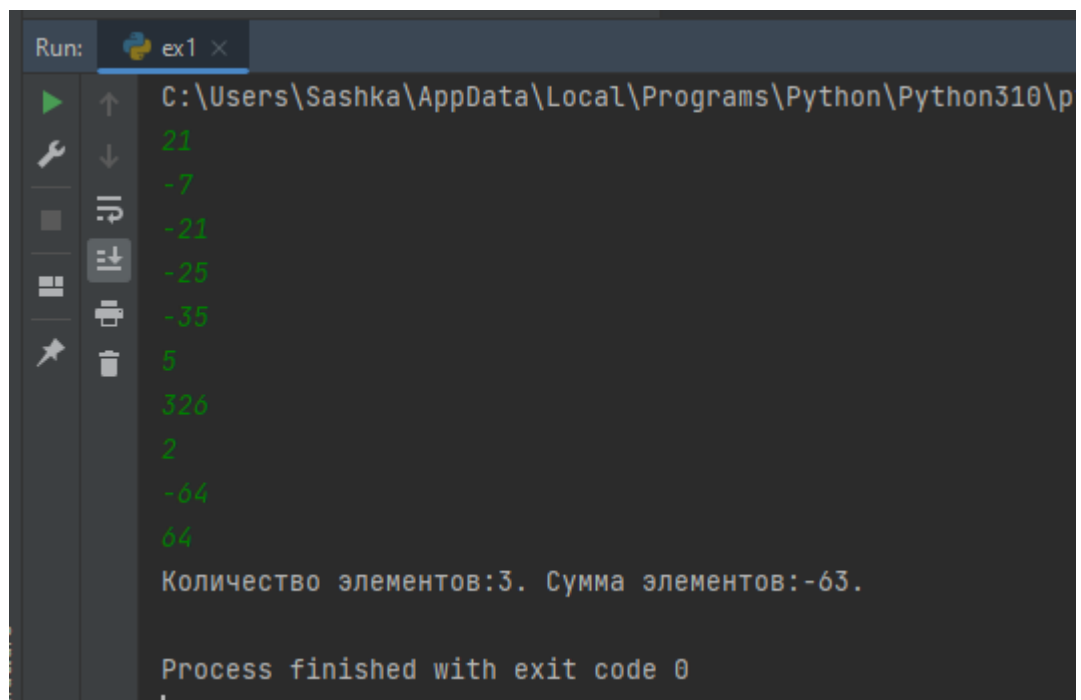
6. Выполнение индивидуального задания №1 (Вариант-11).

Условие: ввести список A из 10 элементов, найти сумму отрицательных элементов кратных 7, их количество и вывести результаты на экран



```
1 #!usr/bin/env python3
2 # -*- coding: utf-8 -*-
3
4 # Вариант 11
5
6 if __name__ == "__main__":
7     A = [int(input()) for _ in range(10)]
8     count = []
9     for i in A:
10         if i < 0 and i % 7 == 0:
11             count.append(i)
12
13     print(f'Количество элементов:{len(count)}. Сумма элементов:{sum(count)}')
```

Рисунок 8 – Код программы



```
Run: ex1 x
C:\Users\Sashka\AppData\Local\Programs\Python\Python310\p
21
-7
-21
-25
-35
5
326
2
-64
64
Количество элементов:3. Сумма элементов:-63.
Process finished with exit code 0
```

Рисунок 9 – Пример выполнения программы

7. Выполнение индивидуального задания №2 (Вариант-11)

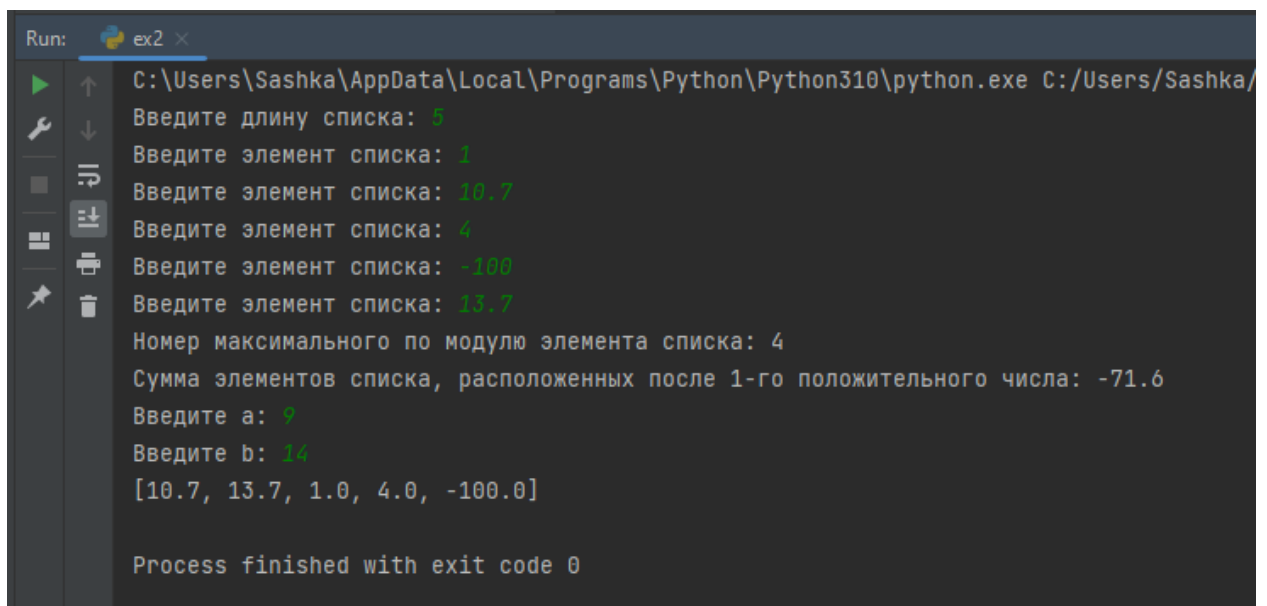
Условие: В списке, состоящем из вещественных элементов, вычислить:

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

```
ex1.py × ex2.py × primer1.py × primer2.py ×
1  ▶  #!usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4      # Вариант 11
5
6  ▶  if __name__ == '__main__':
7      lst = [float(input("Введите элемент списка: "))
8              for i in range(int(input("Введите длину списка: ")))]
9      # решение 1-ой задачи
10     maxx = lst[1]
11     for i in lst:
12         if abs(i) > abs(maxx):
13             maxx = i
14     print(f"Номер максимального по модулю элемента списка: "
15           f"{lst.index(maxx) + 1}")
16
17     # решение 2-ой задачи
18     summ = 0
19     flag = False
20     for i in lst:
21         if flag:
22             summ += i
23         elif i >= 0:
24             flag = True
25     print(f"Сумма элементов списка, расположенных после 1-го"
26           f" положительного числа: {summ}")
27
28     # преобразование списка
29     a, b = int(input("Введите a: ")), int(input("Введите b: "))
30     new_lst = sorted(lst, key=lambda x: int(x) >= a <= b, reverse=True)
31     print(new_lst)
```

Рисунок 10 – Код программы



```
Run: ex2 x
C:\Users\Sashka\AppData\Local\Programs\Python\Python310\python.exe C:/Users/Sashka/
Введите длину списка: 5
Введите элемент списка: 1
Введите элемент списка: 10.7
Введите элемент списка: 4
Введите элемент списка: -100
Введите элемент списка: 13.7
Номер максимального по модулю элемента списка: 4
Сумма элементов списка, расположенных после 1-го положительного числа: -71.6
Введите a: 9
Введите b: 14
[10.7, 13.7, 1.0, 4.0, -100.0]

Process finished with exit code 0
```

Рисунок 11 – Пример выполнения программы

Контрольные вопросы:

1. Что такое списки в языке Python?

О: Списки, это структура данных для хранения объектов различных типов.

2. Как осуществляется создание списка в Python?

О: Список в Python можно создать: `a = [1, 2, 4, 5]`; `a = list()`, `a = []`.

3. Как организовано хранение списков в оперативной памяти?

О: В памяти создается контейнер, который содержит в себе ссылки на объекты элементов массива.

4. Каким образом можно перебрать все элементы списка?

О: Перебрать элементы мы можем при помощи: генераторов, циклов `for`, `while`.

5. Какие существуют арифметические операции со списками?

О: Список можно умножать на число, можно складывать (конкатенация) с другим списком.

6. Как проверить есть ли элемент в списке?

О: При помощи оператора `in`.

7. Как определить число вхождений заданного элемента в списке?

О: Используя метод списков `count`.

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

О: При помощи метода `append` или `insert`.

9. Как выполнить сортировку списка?

О: При помощи метода `sort` или функции `sorted`.

10. Как удалить один или несколько элементов из списка?

О: Можно использовать `del list[a:b]`, методов `pop` и `remove`.

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

О: Списковое включение – это способ построения списков `[i for i in range(10)]`, можно также использовать для обработки, добавив условие.

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

О: `list[start:stop:step]`

13. Какие существуют функции агрегации для работы со списками?

О: функции `min()`, `max()`, `sum()`, `len()`.

14. Как создать копию списка?

О: использовать функцию `copy()` или среза всего списка `[:]`.

15. Самостоятельно изучите функцию `sorted` языка Python. В чем ее отличие от метода `sort` списков?

О: `sort` это метод списков, который меняет уже существующий список, а `sorted` это функция, которая возвращает новый список.