

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ ФГАОУ ВО  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития  
Кафедра инфокоммуникаций

Отчет по лабораторной работе №4

По дисциплине основы кроссплатформенного программирования

«Работа со списками в языке Python»

Выполнила:

студентк группы ИТС-б-о-21-1

Аллаёров Жамшид Хасан угли

---

(подпись)

Проверил: Доцент, к.т.н, доцент  
кафедры

инфокоммуникаций

Воронкин Р. А.

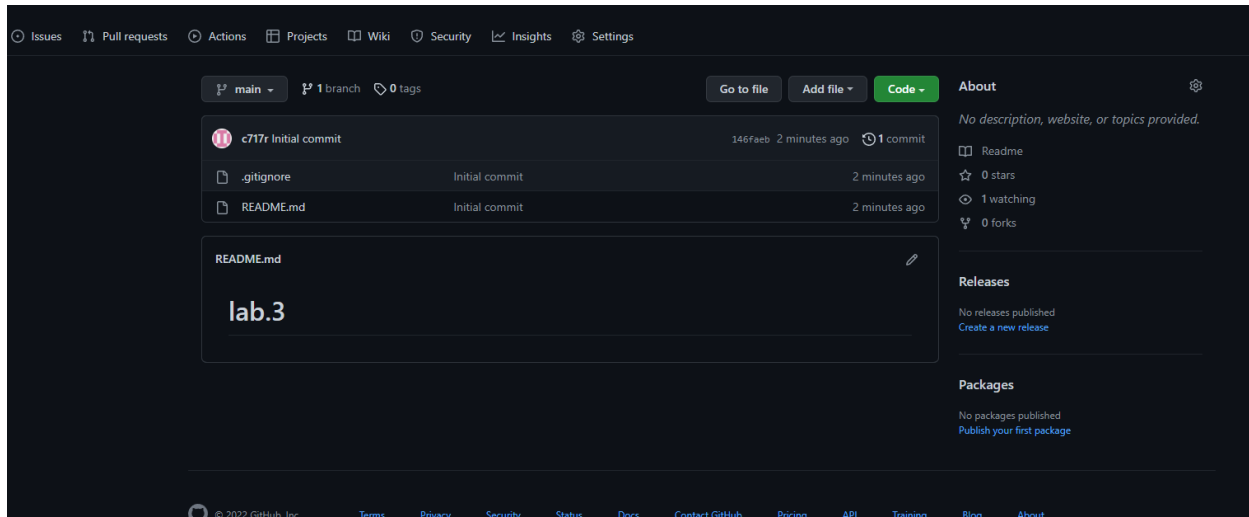
Работа защищена с оценкой:

---

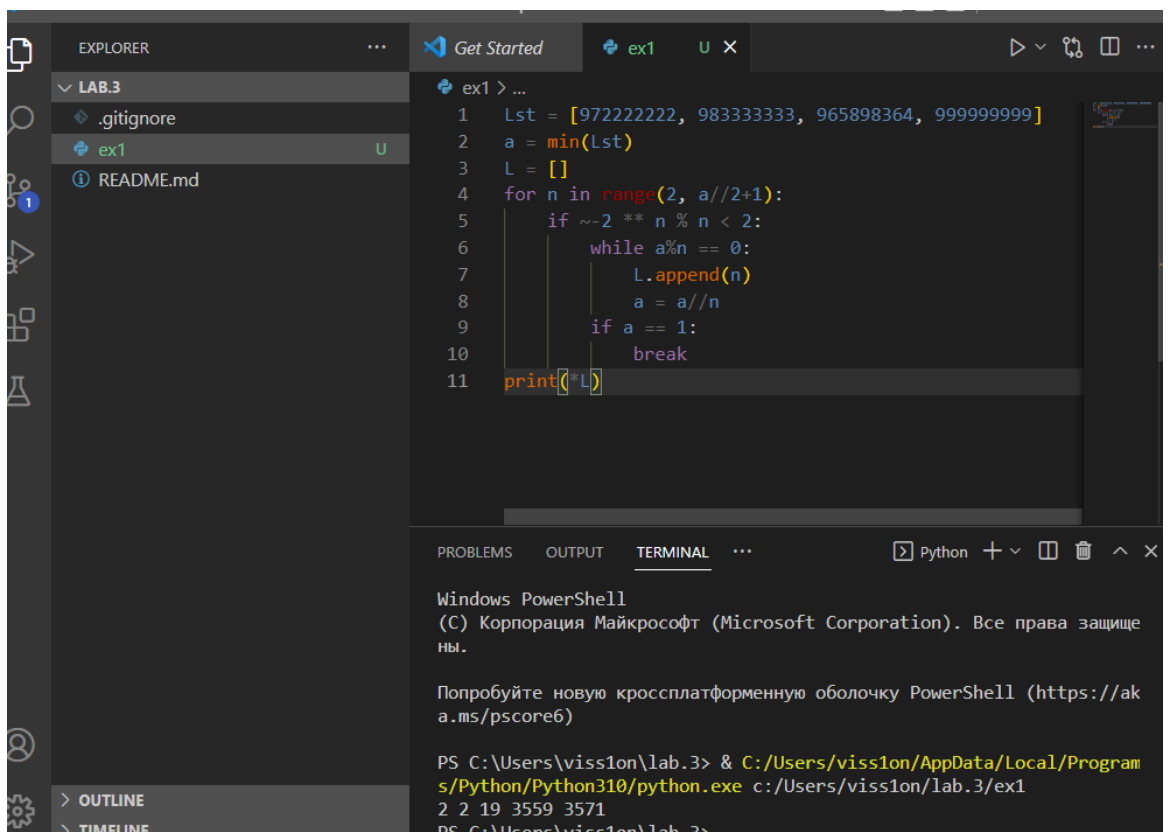
(подпись)

Ставрополь, 2022

Создать общедоступный репозиторий на GitHub, в котором будет использована лицензия MIT и язык программирования Python.



Ввести список  $A$  из 10 элементов, найти наименьший элемент и переставить его с последним элементом. Преобразованный список вывести.



В списке, состоящем из целых элементов, вычислить:

1. произведение элементов списка с четными номерами;
2. сумму элементов списка, расположенных между первым и последним нулевыми элементами.

Преобразовать список таким образом, чтобы сначала располагались все положительные элементы, а потом - все отрицательные (элементы, равные 0, считать положительными).

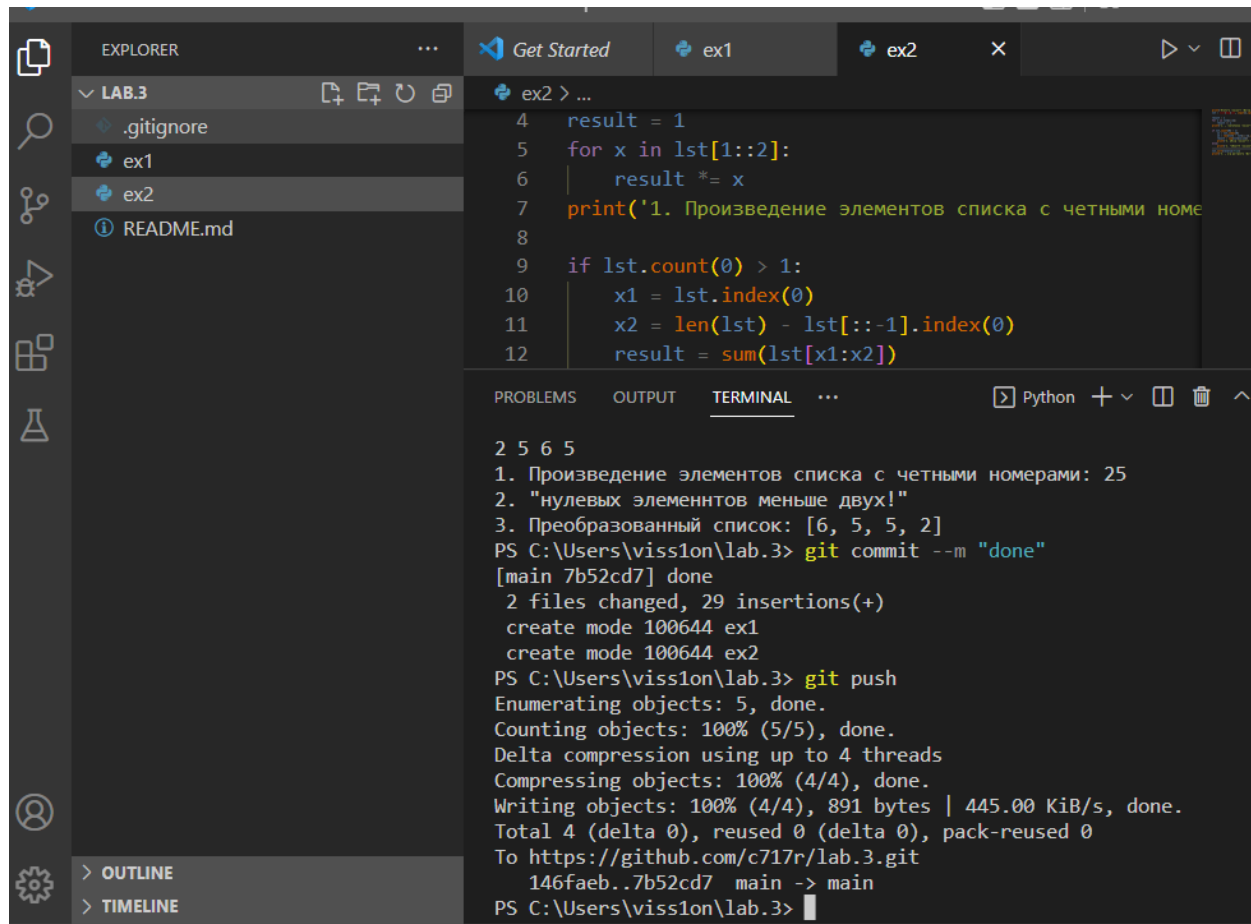
The screenshot shows the Visual Studio Code interface with a Python file named `ex2` open. The Explorer sidebar on the left shows the file structure for `LAB.3`, including `.gitignore`, `ex1`, `ex2`, and `README.md`. The main editor displays the following Python code:

```
4 result = 1
5 for x in lst[1::2]:
6     result *= x
7 print('1. Произведение элементов списка с четными номе
8
9 if lst.count(0) > 1:
10     x1 = lst.index(0)
11     x2 = len(lst) - lst[::-1].index(0)
12     result = sum(lst[x1:x2])
13     print('2. Сумма элементов списка, расположенных ме
14 else:
15     print('2. "нулевых элементов меньше двух! "')
16
17 lst.sort(reverse=True)
18 print('3. Преобразованный список:', lst)
```

At the bottom, the TERMINAL panel shows the execution of the script using Python 3.10. The output is as follows:

```
s/Python/Python310/python.exe c:/Users/viss1on/lab.3/ex2
Введите элементы списка через пробел:
2 5 6 5
1. Произведение элементов списка с четными номерами: 25
2. "нулевых элементов меньше двух!"
3. Преобразованный список: [6, 5, 5, 2]
PS C:\Users\visson\lab.3>
```

Отправьте сделанные изменения на сервер GitHub.



The screenshot shows the Visual Studio Code interface. The Explorer pane on the left shows a project named 'LAB.3' with files: '.gitignore', 'ex1', 'ex2', and 'README.md'. The file 'ex2' is selected. The editor shows the following Python code:

```
4 result = 1
5 for x in lst[1::2]:
6     result *= x
7 print('1. Произведение элементов списка с четными номе
8
9 if lst.count(0) > 1:
10     x1 = lst.index(0)
11     x2 = len(lst) - lst[::-1].index(0)
12     result = sum(lst[x1:x2])
```

The TERMINAL pane at the bottom shows the output of running the script:

```
2 5 6 5
1. Произведение элементов списка с четными номерами: 25
2. "нулевых элементов меньше двух!"
3. Преобразованный список: [6, 5, 5, 2]
PS C:\Users\viisslon\lab.3> git commit --m "done"
[main 7b52cd7] done
2 files changed, 29 insertions(+)
create mode 100644 ex1
create mode 100644 ex2
PS C:\Users\viisslon\lab.3> git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 891 bytes | 445.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/c717r/lab.3.git
146faeb..7b52cd7 main -> main
PS C:\Users\viisslon\lab.3>
```

## 1. Что такое списки в языке Python?

Списки в Python представляют собой упорядоченные изменяемые наборы объектов, пронумерованных от 0. При этом объекты могут быть разными — от целых чисел до строк. Списки могут также хранить в себе списки.

## 2. Как осуществляется создание списка в Python?

Списки в Python представляют собой упорядоченные изменяемые наборы объектов, пронумерованных от 0. При этом объекты могут быть разными — от целых чисел до строк. Списки могут также хранить в себе списки.

В статье разберёмся с базовыми принципами списков в Питоне, а также рассмотрим методы работы с ними. Если вы изучаете Python с нуля, предлагаем также ознакомиться с дорожной картой для начинающих.

Хранение в памяти

Создание списка

Срезы (slice)

Простые операции

Методы списков

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

Python — это интерпретируемый язык программирования, поэтому перед запуском программы код на языке Python компилируется в машиночитаемые инструкции — [байт-код](#). Инструкции байт-кода интерпретируются виртуальной машиной, определяемой реализацией языка, например, стандартной — [CPython](#).

Оговоримся, что CPython не взаимодействует напрямую с регистрами и ячейками физической памяти — только с ее **виртуальным представлением**. В начале выполнения программы операционная система создает новый процесс и выделяет под него ресурсы. Выделенную виртуальную память интерпретатор использует для 1) собственной корректной работы, 2) стека вызываемых функций и их аргументов и 3) хранилища данных, представленного в виде [кучи](#).

В отличие от C/C++, мы не можем управлять состоянием кучи напрямую из Python. Функции низкоуровневой работы с памятью предоставляются [Python/C API](#), но обычно интерпретатор просто обращается к хранилищу данных через **диспетчер памяти Python** (memory manager).

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

Цикл **for** в языке программирования Python предназначен для перебора элементов структур данных и некоторых других объектов. Это не цикл со счетчиком, каковым является **for** во многих других языках. Что

значит перебор элементов? Например, у нас есть список, состоящий из ряда элементов. Сначала берем из него первый элемент, затем второй, потом третий и так далее. С каждым элементом мы выполняем одни и те же действия в теле `for`. Нам не надо извлекать элементы по их индексам и заботиться, на каком из них список заканчивается, и следующая итерация бессмысленна. Цикл `for` сам переберет и определит конец.

## 5. Какие существуют арифметические операции со списками?

Списки бывают полезны при создании баз знаний (баз данных), экспертных систем, словарей; перечень областей применения можно продолжать еще долго. В настоящей главе рассматриваются структура, организация и представление списков, демонстрируются некоторые из методов, применяемых при программировании на Турбо-Прологе.

Список является набором объектов одного и того же доменного типа. Объектами списка могут быть целые числа, действительные числа, символы, символьные строки и структуры. Порядок расположения элементов является отличительной чертой списка; те же самые элементы, упорядоченные иным способом, представляют уже совсем другой список. Порядок играет важную роль в процессе сопоставления.

Турбо-Пролог допускает списки, элементами которых являются структуры. Если структуры принадлежат к альтернативному домену, элементы списка могут иметь разный тип. Такие списки используются для специальных целей.

Для удобства обработки списков в Прологе введены два понятия: голова и хвост. Первый элемент списка (или несколько первых элементов) являются головой списка, а оставшиеся - его хвостом

Элементы списка разделяются запятыми и заключаются в квадратные скобки. Любой список представляет собой:

- либо пустой список (атом `[]`);
- либо непустой список - структуру, состоящую из двух частей:

- первый элемент - голова (Head) списка;
- второй элемент - хвост (Tail) списка.

В общем случае голова списка может быть любым объектом языка Пролог, а хвост - обязательно должен быть списком. Поскольку хвост - список, то он либо пуст, либо имеет свои собственные голову и хвост.

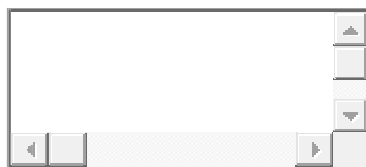
#### 6. Как проверить есть ли элемент в списке?

Простой и рудиментарный метод проверки, содержит ли список элемент, проходит через него и проверяет, соответствует ли элемент, на котором мы находимся, тому, который мы ищем. Давайте используем для этого цикл `for` :



```
for animal in animals:  
    if animal == 'Bird':  
        print('Chirp!')
```

Этот код приведет к:



Chirp!

#### 7. Как определить число вхождений заданного элемента в списке?

Метод `python count()` подсчитывает количество вхождений элемента в списке и возвращает найденное значение.

Синтаксис:

```
list.count(x)
```

Метод `count()` принимает один аргумент `x`, значение которое нужно найти. Данный метод возвращает количество вхождений элемента в список.

Пример:

# объявление списка

```
website_list = ['google.com', 'includehelp.com', 'linkedin.com', 'google.com']
```

# подсчет вхождений 'google.com'

```
count = website_list.count('google.com')
```

```
print('google.com found', count, 'times.')
```

# подсчет вхождений 'linkedin.com'

```
count = website_list.count('linkedin.com')
```

```
print('linkedin.com found', count, 'times.')
```

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

Тип данных списка Python имеет три метода для добавления элементов:

- `append()` — добавляет один элемент в список.
- `extend()` — добавляет элементы итерируемого в список.
- `insert()` — вставляет один элемент в заданную позицию списка.

Все три метода изменяют список на месте и возвращают `None`.

Метод `append()` в списке Python

Метод `append()` добавляет один элемент в конец списка.

Синтаксис метода `append()` следующий:

```
list.append(element)
```

Где, элемент `element`, который будет добавлен в список.

Вот пример:

```
characters = ['Tokyo', 'Lisbon', 'Moscow', 'Berlin']
```

```
characters.append('Nairobi')
```



```
print('Updated list:', characters)
```

#### 9. Как выполнить сортировку списка?

Выполните указанные ниже действия:

Выделите столбцы для сортировки. ...

На ленте выберите Данные > Сортировка.

Во всплывающем окне Сортировка в

раскрывающемся списке Сортировать по выберите столбец, по которому нужно выполнить сортировку. ...

В раскрывающемся списке Порядок выберите Настраиваемый список.

#### 10. Как удалить один или несколько элементов из списка?

Тип данных List в Python помогает сохранять разные типы данных в определенной связанной последовательности. Данные записываются в квадратные скобки и разделяются запятыми.

1. В Python есть несколько методов для удаления элементов из списка: `remove()`, `pop()` и `clear()`. Помимо них также существует ключевое слово `del`.

2. Рассмотрим их все.

3. Пример списка:

4. КОПИРОВАТЬ

5. `my_list = ['Python', 50, 11.50, 'Alex', 50, ['A', 'B', 'C']]`

6. Индекс начинается с 0. В списке `my_list` на 0-ой позиции находится строка «Python». Далее:

7. Целое число 50

8. Число с плавающей точкой 11.50

9. Снова строка — «Alex»

10. Еще одно число 50

11. Список из строк «А», «В» и «С»