

Санкт-Петербургский Национальный Исследовательский Университет  
Информационных технологий, механики и оптики

Факультет инфокоммуникационных технологий

**Лабораторная работа №3**  
**Вариант №1**

Выполнили:

Смирнов И.И.

Касьяненко В.М.

Проверил:

Мусаев А.А.

Санкт-Петербург

2022

## СОДЕРЖАНИЕ

Введение.....	3
1 Задание 1 .....	4
1.1 Сортировка .....	4
2 Задание 2 .....	5
2.1 Реализация алгоритмов различной сложности.....	5
3 Задание 3 .....	6
3.1 Построение зависимости алгоритмов различной сложности .....	6
Заключение .....	8
Список литературы .....	9
Приложение .....	10

## ВВЕДЕНИЕ

Для становления хорошим специалистом в области программирования на языке Python необходимо знать основной функционал языка.

Цель данной работы – ознакомление с сортировкой и оцениванием сложности алгоритма на языке программирования Python.

В ходе выполнения лабораторной работы будут решены следующие задачи:

- ознакомление с сортировкой;
- реализация алгоритмов, имеющих различную сложность;
- построение зависимости между количеством элементом и количеством шагов для алгоритмов с различной сложностью.

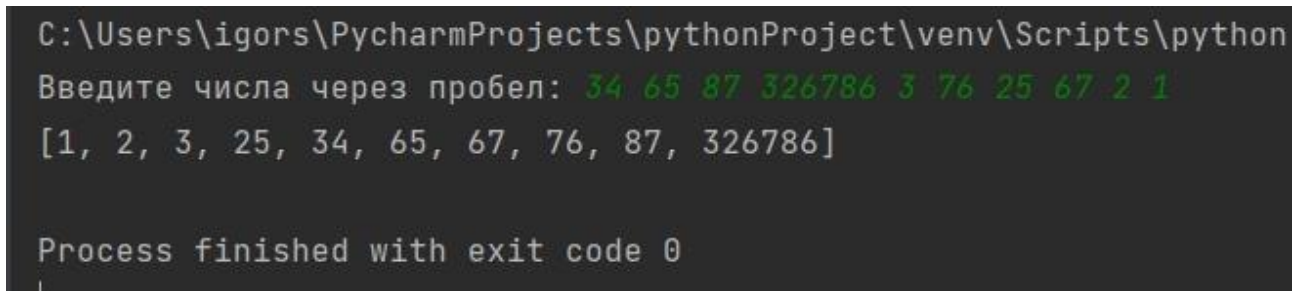
Задания, которые необходимо выполнить:

1. Задание 1: написать программу для пузырьковой сортировки, оценить сложность данного метода, сравнить с методом `sort()`.
2. Задание 2: придумать и реализовать алгоритмы, имеющие сложность  $O(3n)$ ,  $O(n \log n)$ ,  $O(n!)$ ,  $O(n^3)$ ,  $O(3 \log(n))$ ;
3. Задание 3: построить зависимость между количеством элементом и количеством шагов для алгоритмов со сложностью  $O(1)$ ,  $O(\log n)$ ,  $O(n^2)$ ,  $O(2^n)$ , сравнить сложность данных алгоритмов.

## 1 ЗАДАНИЕ 1

### 1.1 Сортировка

Была написана программа пузырьковой сортировки, в которой на каждом шаге находится наибольший элемент из двух соседних. Этот элемент ставится в конец пары. Таким образом, происходит эта сортировка (рисунок 1).



```
C:\Users\igors\PycharmProjects\pythonProject\venv\Scripts\python
Введите числа через пробел: 34 65 87 326786 3 76 25 67 2 1
[1, 2, 3, 25, 34, 65, 67, 76, 87, 326786]

Process finished with exit code 0
```

Рисунок 1 – Вывод программы с пузырьковой сортировкой

Данный алгоритм пузырьковой сортировки имеет сложность  $O(n^2)$ . В то время, как встроенный в Python метод `sort()` предназначен для более широкого круга решения задач и имеет сложность  $O(n \log n)$ . То есть написанная программа имеет большую сложность.

## 2 ЗАДАНИЕ 2

### 2.1 Реализация алгоритмов различной сложности

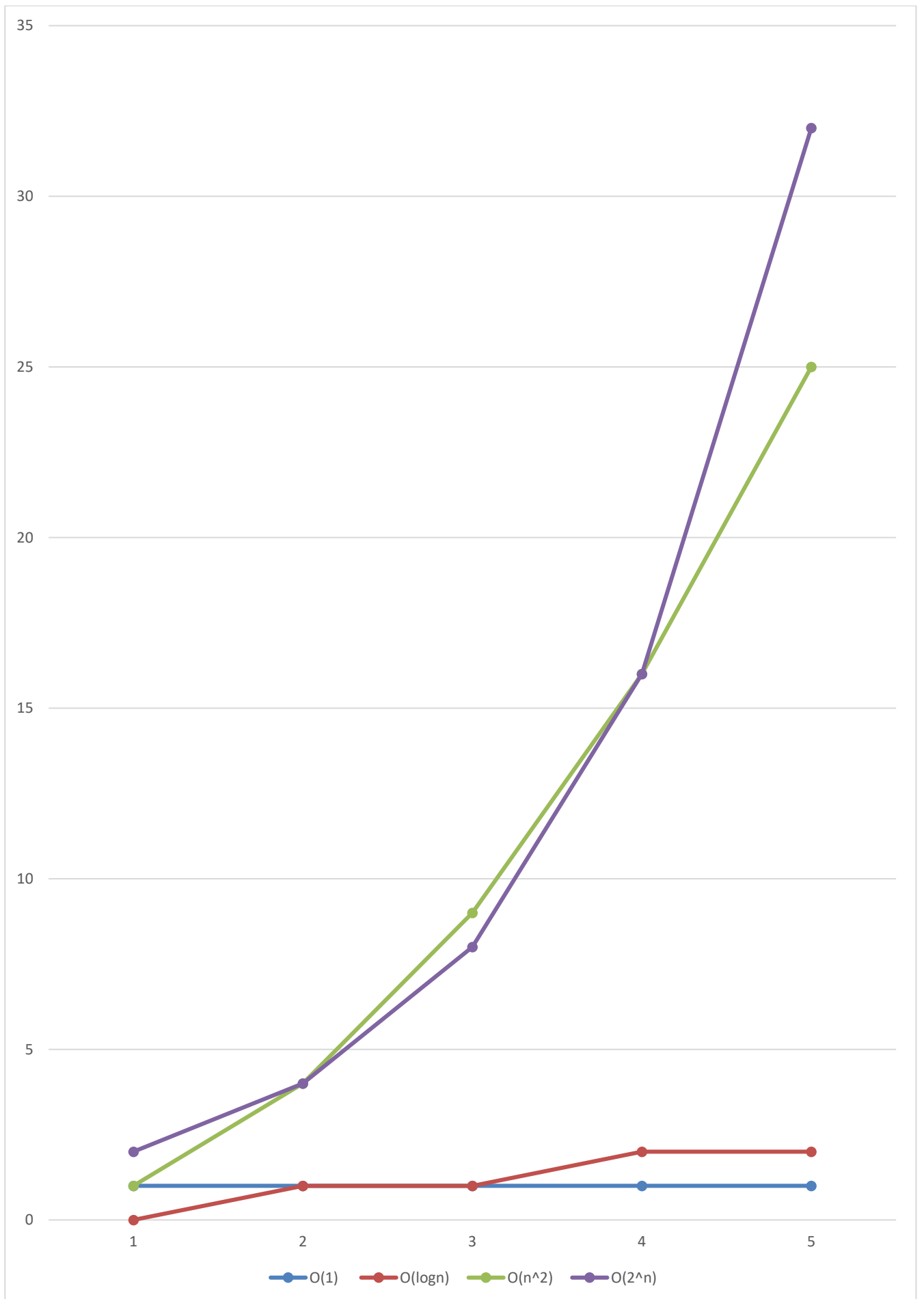
- 1) Алгоритм сложности  $O(3n)$  можно построить при помощи использования цикла `for` и сложения 3 элементов в нем.
- 2) Алгоритм сложности  $O(n \log n)$  можно получить при применении сортировки слиянием (числа разбиваются на 2 части, и каждая часть сортируется отдельно, после чего эти части объединяются).
- 3) Алгоритм сложности  $O(n!)$  можно получить, например, если составлять всевозможные уникальные комбинации из заданных элементов.
- 4) Алгоритм сложности  $O(n^3)$  можно построить при помощи трех вложенных друг в друга циклов `for`.
- 5) Алгоритм сложности  $O(3 \log(n))$  можно получить при использовании бинарного поиска, каждый раз складывая 3 элемента в конце при выполнении цикла.

### 3 ЗАДАНИЕ 3

#### 3.1 Построение зависимости алгоритмов различной сложности

Построим зависимость между количеством элементом и количеством шагов для алгоритмов с различной сложностью в виде таблицы и графиков. Сравнив все алгоритмы, можно заметить, что самый сложным алгоритмом является  $O(2^n)$ , далее  $O(n^2)$ , затем  $O(\log n)$  и потом  $O(1)$ .

Количество шагов	$O(1)$	$O(\log n)$	$O(n^2)$	$O(2^n)$
1	1	0	1	2
2	1	1	4	4
3	1	1	9	8
4	1	2	16	16
5	1	2	25	32
6	1	2	36	64
7	1	2	49	128
8	1	3	64	256
9	1	3	81	512
10	1	3	100	1024
11	1	3	121	2048
12	1	3	144	4096
13	1	3	169	8192
14	1	3	196	16384
15	1	3	225	32768
16	1	4	256	65536



## **ЗАКЛЮЧЕНИЕ**

В ходе лабораторной работы был получен опыт написания программы для пузырьковой сортировки, а также опыт построения и сравнения алгоритмов различной сложности.



## СПИСОК ЛИТЕРАТУРЫ

1. Wikipedia. Временная сложность алгоритма. [Электронный ресурс] – [https://ru.wikipedia.org/wiki/%D0%92%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F\\_%D1%81%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D1%8C\\_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B0](https://ru.wikipedia.org/wiki/%D0%92%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B0) (Дата последнего обращения 27.10.2022).

## ПРИЛОЖЕНИЕ

Ссылка на полный код данной лабораторной работы:

[https://github.com/VeraKasianenko/Algorithms\\_1\\_term\\_ICT/tree/main/labs/lab\\_3](https://github.com/VeraKasianenko/Algorithms_1_term_ICT/tree/main/labs/lab_3)