Санкт-Петербургский Национальный Исследовательский Университет

Информационных Технологий, Механики и Оптики

Факультет информационных технологий

Лабораторная работа №2

Выполнили:

Лазуренко А.В., Маатук А.

Проверил

Мусаев А.А.

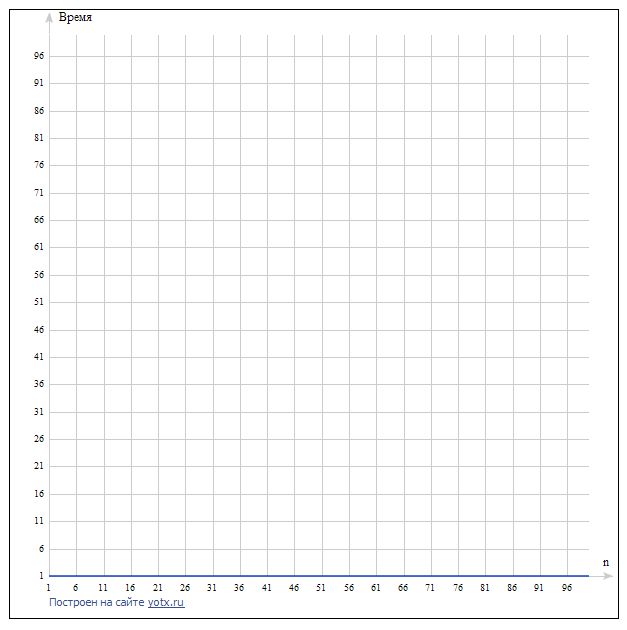
Санкт-Петербург,

2023

ЗАДАНИЕ 1

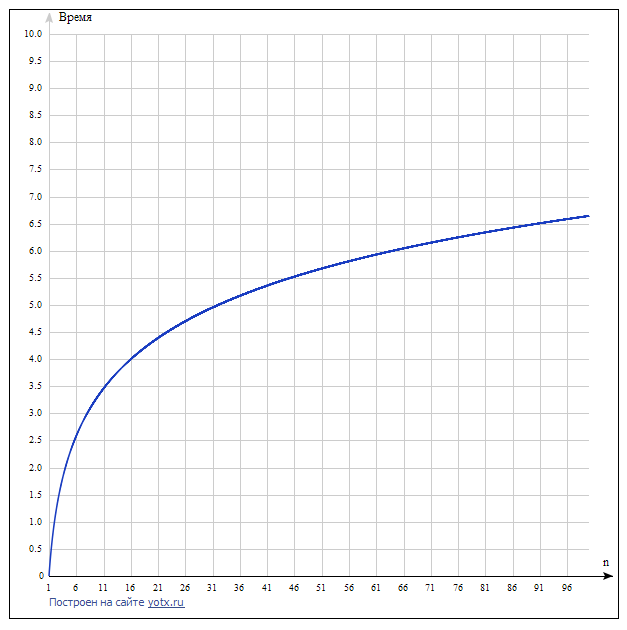
Для построения зависимости между количеством элементов и количеством шагов для алгоритмов со сложностью O(1), O(logn), O(n^2), O(2^n), мы можем использовать графики.

1. Сложность O(1): Алгоритмы с постоянной сложностью имеют фиксированное число шагов, независимо от размера входных данных. Поэтому в данном случае зависимость будет представлена горизонтальной линией на графике.



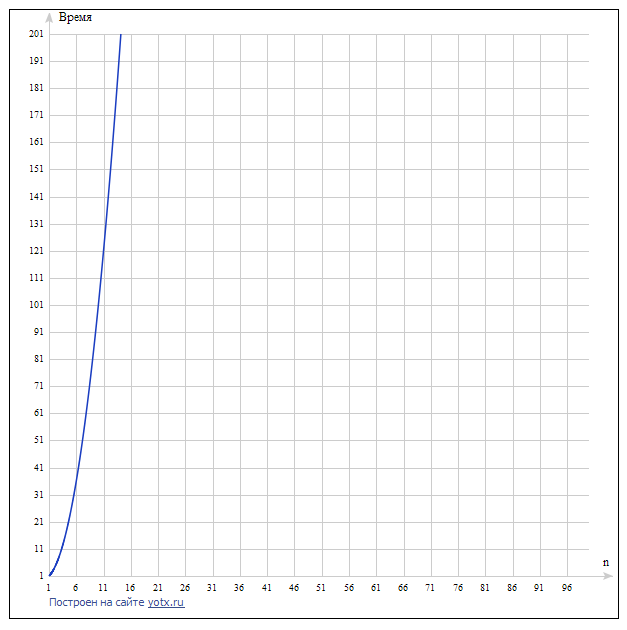
*Рисунок 1 -График зависимости для сложности О(1)*

2. Сложность O(logn): Алгоритмы с логарифмической сложностью растут медленно, при увеличении размера входных данных. Зависимость будет представлять собой постепенно возрастающую кривую.



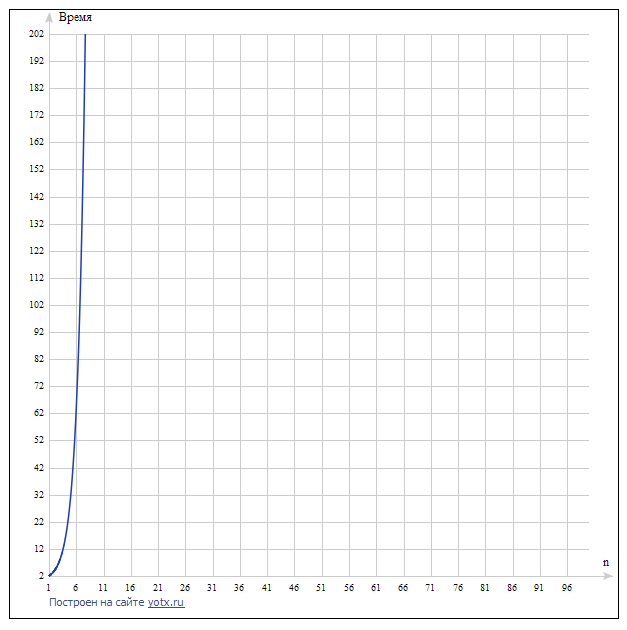
*График зависимости для сложности O(logn)*

3. Сложность O(n^2): Алгоритмы со квадратичной сложностью растут быстро, при увеличении размера входных данных. Зависимость будет представлять собой параболу, с вершиной в начале координат.



*График зависимости для сложности O(n^2)*

4. Сложность O(2^n): Алгоритмы с экспоненциальной сложностью растут очень быстро, при увеличении размера входных данных. Зависимость будет представлять собой стремительно возрастающую экспоненту.

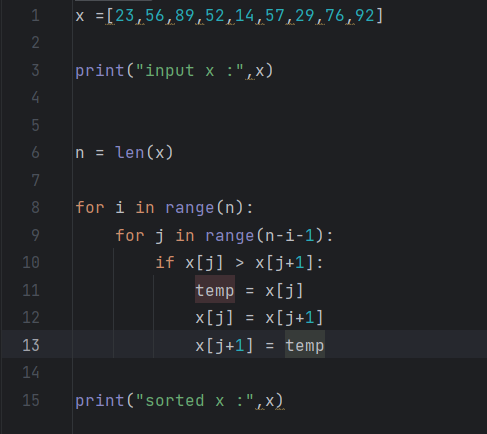


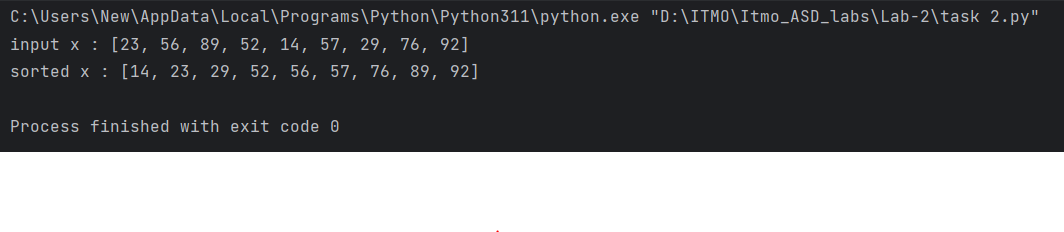
*График зависимости для сложности O(2^n)*

Сравнение сложности данных алгоритмов показывает, что алгоритмы с константной и логарифмической сложностью наиболее эффективны, а алгоритмы с квадратичной и экспоненциальной сложностью являются наиболее затратными при обработке больших объемов данных.

ЗАДАНИЕ 2

Сложность алгоритма Пузырьковой сортировки составляет



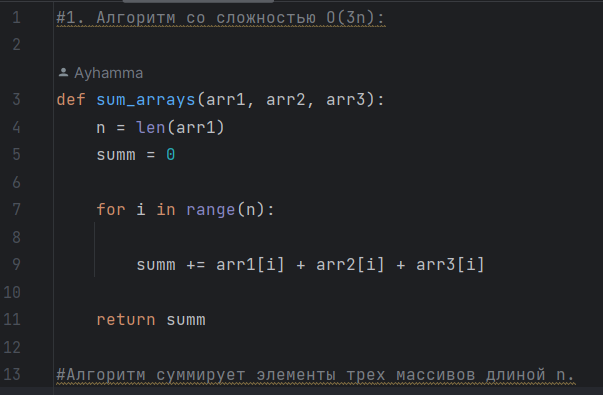


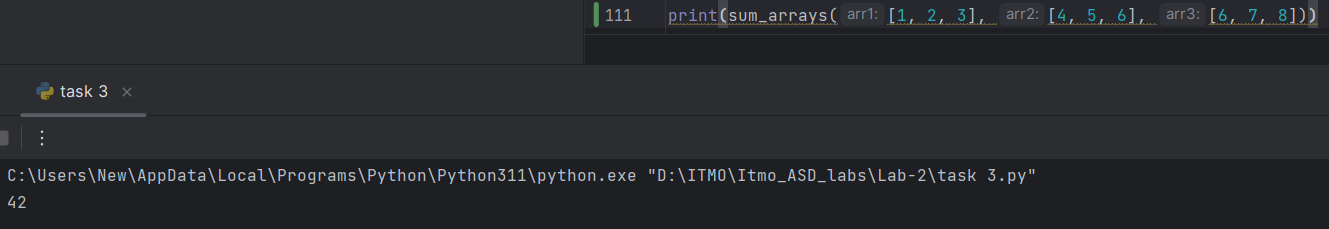
*Программа к заданию 2 и результат выполнения*

В Python используется алгоритм Timsort — гибридный алгоритм сортировки, сочетающий сортировку вставками и сортировку слиянием. Его сложность составляет O(n log n) в худшем случае и O(n) – в лучшем. Этот алгоритм намного эффективнее алгоритма сортировки пузырьком

ЗАДАНИЕ 3

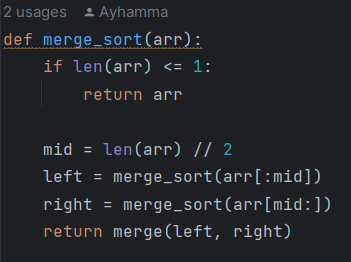
В данном задании реализовали следующие алгоритмы



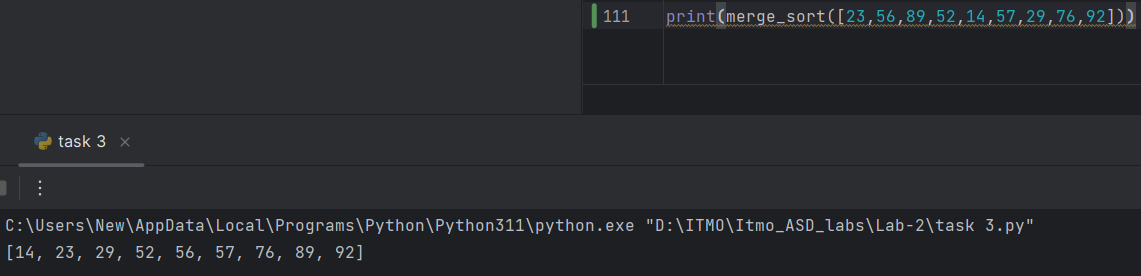


*Алгоритм сложности O(3n) и результат его выполнения*

Так как выполняется n команд, каждая по 3 раза, то сложность алгоритма соответствует заданию.

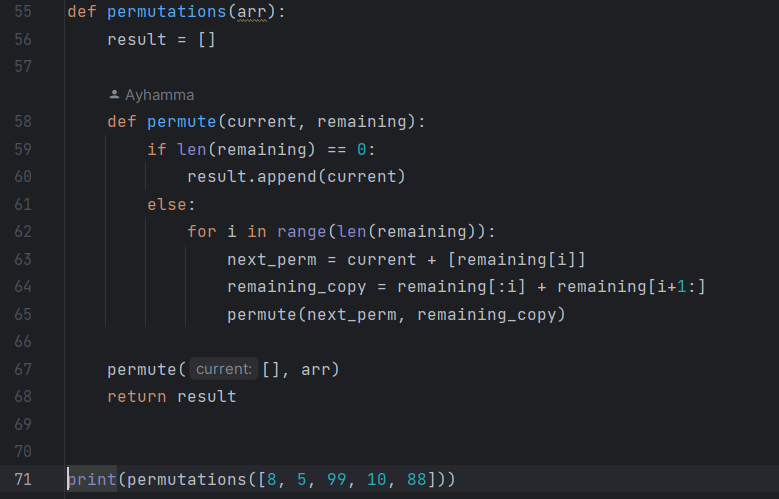


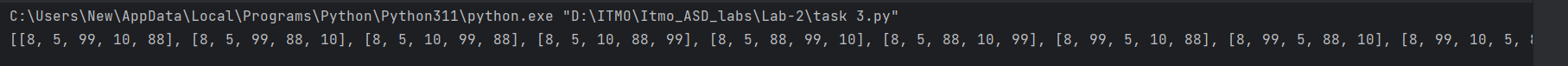




*Алгоритм сложности O(n log(n)) и результат его выполнения*

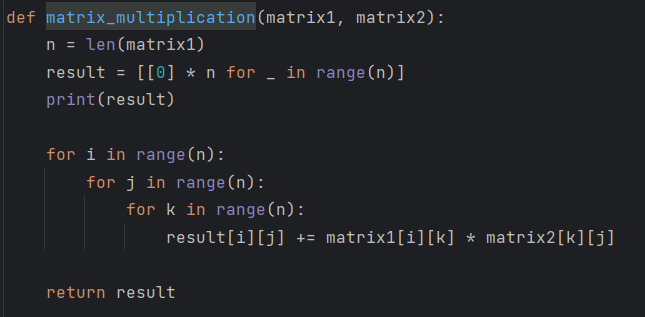
Так как исходный список делится на пополам в зависимости от его длины n, а сам алгоритм разбиения будет вызываться около n раз, то сложность алгоритма составляет O(n log(n))





*Алгоритм сложности O(n!) и результат его выполнения*

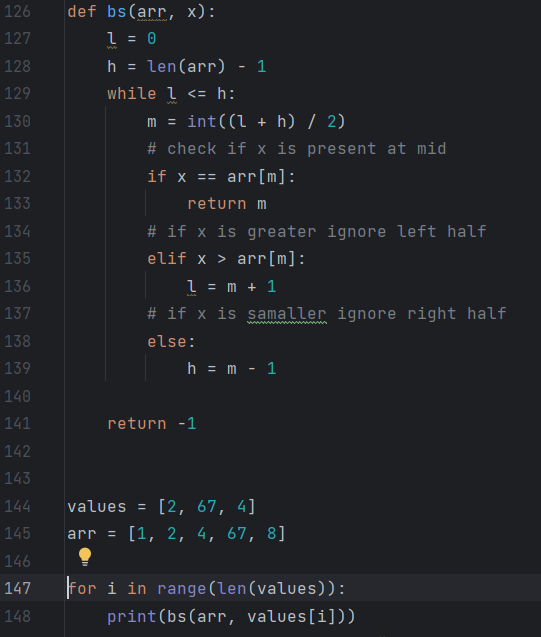
В данном алгоритме выполняется рекурсия для каждого элемента списка, поэтому сложность будет составлять O(n!).

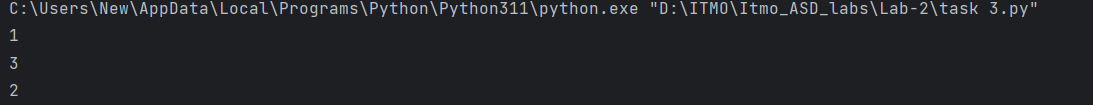




*Алгоритм сложности O(n^3) и результат его выполнения*

Так как происходит перебор n элементов в трех вложенных циклах, сложность алгоритма составляет O(n^3).





*Алгоритм сложности O(3log(n)) и результат его выполнения*

В этой программе реализован алгоритм бинарного поиска, сложность которого составляет O(log n). Входной список делится на 2^n частей, поэтому сложность возрастает логарифмически. Так как производится поиск трех элементов, можно сказать, что сложность алгоритма составляет O(3log n).

Использемые источники

1. yotx.ru. Построение графиков функции онлайн [Электронный ресурс] –

<https://yotx.ru/>

(Дата обращения 01.10.2023)