Санкт-Петербургский Национальный Исследовательский Университет

Информационных Технологий, Механики и Оптики

Факультет инфокоммуникационных технологий

Лабораторная работа №3

Выполнили:

Конопля А.К., Горлов И.В., Язев Г.А.

Проверил:

Мусаев А.А.

Санкт-Петербург

2022

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc118404109)

[ПОСТАНОВКА ЗАДАЧИ 4](#_Toc118404110)

[Ход работы 5](#_Toc118404111)

[ЗАКЛЮЧЕНИЕ 7](#_Toc118404112)

[СПИСОК ЛИТЕРАТУРЫ 8](#_Toc118404113)

[ПРИЛОЖЕНИЯ 9](#_Toc118404114)

ВВЕДЕНИЕ

Данная работа нацелена на изучение понятия сложности алгоритма и построение алгоритмов определённой сложности.

ПОСТАНОВКА ЗАДАЧИ

Задача данной лабораторной работы изучение и работа с понятием сложность алгоритмов на языке Python.

Ход работы

**Задание 1. Пузырьковая сортировка.**

Алгоритм основан на последовательном переборе и перестановке элементов последовательности. Данный алгоритм имеет сложность O(n2).

Вывод: было изучено понятие пузырьковой сортировки и написана программа, на языке Python, реализующая пузырьковую сортировку.

**Задание 2. Придумать и реализовать алгоритмы, разной сложностиA.**

В ходе выполнения данного задания были реализованы алгоритмы сложности O(3n), O(n\*log2(n)), O(n!), O(n­3), O(3\*log2(n)). алгоритм сложности O(3n) является простым перебором чисел, и записью в массив условных значений, данная программа не несёт в себе смысловой нагрузки, но в ней реализован алгоритм сложности O(3n). Алгоритм O(n\*log2(n)) представляет собой «умную сортировку», встроенную в Python. Алгоритм O(n!) представлен как функция для нахождения факториала числа. Алгоритмом сложности O(n3) является 3 цикла, длинной n, данная программа не несёт в себе смысловой нагрузки, но является алгоритмом сложности O(n3). Алгоритм сложности O(3\*log2(n)) представляет из себя тройной бинарный поиск, сам бинарный поиск имеет сложность O(log­2(n)).

Вывод: было подробнее изучено понятие сложности алгоритмов, были реализованы алгоритмы различной сложности.

**Задание 3. Построить зависимость между количеством элементов и количеством шагов.**

Были построены графики (см. приложение B).

Количество шагов алгоритма O(1) не зависит от количества переменных. Это самый простой алгоритм. Количество шагов алгоритма O(log(n)) возрастает пропорционально количеству элементов, но растёт значительно медленнее, это делает его простым алгоритмом. Количество шагов алгоритма O(n2), в сравнении с количеством элементов, возрастает значительно быстрее, этот алгоритм значительно сложнее чем предыдущие. График количества шагов от количества элементов возрастает быстрее всего у алгоритма O(2n), экспоненциальная сложность алгоритма является одной из самых сложных.

Вывод: наиболее простой алгоритм из представленных – O(1), константный алгоритм, количество шагов в котором не зависит от количества элементов. Самый сложный алгоритм O(2n) – экспоненциальный, график сложности этого алгоритма возрастает быстрее других. Были изучены разные сложности алгоритмов, а также проведено сравнение сложности алгоритмов.

ЗАКЛЮЧЕНИЕ

В ходе данной работы было изучено понятие сложности алгоритмов и написаны программы, на языке Python, реализующие алгоритмы различной сложности.

СПИСОК ЛИТЕРАТУРЫ

1. Proglib. Сложность алгоритмов и операций на примере Python. [Электронный ресурс]. [Сайт] URL - <https://proglib.io/p/slozhnost-algoritmov-i-operaciy-na-primere-python-2020-11-03>.

ПРИЛОЖЕНИЯ

1. Задание 3. Алгоритмы и структуры данных. Конопля Горлов Язев.xlsx
2. <https://github.com/AlexeyKonoplia/Lab_3_alg_Konoplya_Gorlov_Yazev>