Санкт-Петербургский Национальный Исследовательский Университет

Информационных Технологий, Механики и Оптики

Факультет инфокоммуникационных технологий

Лабораторная работа №4

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

В данной работе необходимо изучить и написать ан языке python, алгоритмы сортировок: сортировка «расчёской», быстрая сортировка, блочная сортировка, пирамидальная сортировка, а также оценить достоинства, недостатки и сложность алгоритмов.

ПОСТАНОВКА ЗАДАЧИ

Задача лабораторной работы: реализовать алгоритмы сортировок: сортировка «расчёской», быстрая сортировка, блочная сортировка, пирамидальная сортировка, а также оценить достоинства, недостатки и сложность алгоритмов.

Ход работы

**1 Задание. Быстрая сортировка. Сортировка расчёской.**

**Быстрая сортировка.** В ходе выполнения первого задания был реализован алгоритм быстрой сортировки, основанный на разделении списка на две части, в одной из которых находятся элементы равные или большие чем какой-либо выбранный элемент (в данном случае мы берём элемент из центра списка), а в другой меньшие.

**Сортировка расчёской.** В ходе выполнения перового задания был реализован алгоритм сортировки расчёской, данный метод основан на прохождении массива и сравнении 2х элементов, но ен соседних, как в случае с пузырьковой сортировкой, а стоящих на некотором удалении “на краях расчёски”. В конце выполнения алгоритма длина расчёски уменьшается до 1 и начинается пузырьковая сортировка. Данный метод позволяет устранить маленькие числа в конце списка, которые значительно замедляют сортировку пузырьком.

**2 Задание. Блочная сортировка. Пирамидальная сортировка.**

**Блочная сортировка.** В ходе выполнения второго задания был реализован алгоритм блочной сортировки, данный метод основан на разделении списка на отдельные блоки и сортировки элементов внутри этого блока. Количество блоков, на которые необходимо разделить массив минимально. В данном методе также был использован метод сортировки вставкой, использующийся для сортировки в блоках, основанный на вставке следующего элемента между предыдущими, с помощью сравнения.

**Пирамидальная сортировка.** В ходе выполнения второго задания был реализован алгоритм блочной сортировки, данный метод основан на бинарном дереве, где к каждому элементу списка приписывается 2 младших элемента и так далее. В ходе сортировки берётся элемент массива и сравнивается с 2-мя младшими элементами, больший из них поднимается наверх и так далее, пока элементы не будут упорядочены так, что самый старший элемент — это наибольшее число.

**3 Задание. Оценка алгоритмов.**

**Быстрая сортировка.**

Достоинства:

* Один из самых быстродействующих методов.
* Прост в исполнении

Недостатки:

* Деградирует по скорости до O(n2) при худшем случае
* Неустойчив, т.е. алгоритм меняет относительный порядок сортируемых элементов, важный элемент при сортировке списков, состоящих из кортежей.

Среднее время O(n\*log n)

Лучшее время O(n)

Худшее время O(n2)

**Сортировка расчёской**

Достоинства:

* Меньшее, по сравнению с пузырьковой сортировкой, количество перестановок. Следовательно меньшая пространственная сложность

Недостатки:

* Неустойчив
* Худшая сложность n2

Лучшее время алгоритма O(n\*log n)

Среднее время O(n2 / 2p), где p – количество инкрементов, то есть количество увеличивающихся элементов. (пример x += 1)

Худшее время O(n2)

**Блочная сортировка**

Достоинства:

* Быстрый метод, среднее время O(n)

Недостатки:

* Деградирует до O(n2) при неудачном алгоритме получения номера корзины или большом количестве мало отличных элементов.
* Сложный для реализации алгоритм, особенно для новчков

Лучшее / среднее время O(n)

Худшее время O(n2)

**Пирамидальная сортировка**

Достоинства:

* Даже в худшем случае сложность алгоритма O(n \* log n)

Недостатки:

* В лучшем случае сложность алгоритма O(n \* log n)
* Такая сложность алгоритма имеет преимущество либо в худших случаях, либо при большом количестве элементов.

Лучшее / Среднее / Худшее время O(n \* log n)

ЗАКЛЮЧЕНИЕ

В ходе данной работы были изучены, написаны и оценены алгоритмы: быстрой сортировки, сортировки расчёской, блочной сортировки и пирамидальной сортировки.

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. <https://github.com/AlexeyKonoplia/lab_4_Konoplia_Gorlov_Yazev>