МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ВЫСШАЯ ШКОЛА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И

ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ

ОТЧЕТ

|  |  |
| --- | --- |
| ФИО студента | Подлевских Дмитрий Михайлович |
| Группа | 11-808 |

Казань 2019

1. **Shell Sort. Сортировка Шелла.**

**История:**

**Сортировка Шелла была названа в честь её изобретателя — Дональда Шелла, который опубликовал этот алгоритм в 1959 году.**

**Дональд Шелл:**

**После окончания Мичиганского технологического университета Дональд Шелл поступил в Инженерные войска США. После окончания войны он вернулся в Мичиган и начал преподавать. Затем Шелл переехал в Цинциннати и начал работу в General Electric, где разрабатывал алгоритмы и написал программу для выполнения круговых расчётов для двигателей самолётов. В то же время он учился в университете Цинциннати и в 1951 году защитил магистерскую диссертацию, а затем в 1959 году стал доктором философии. В июле того же года он опубликовал свой алгоритм сортировки.**

1. **Принцип работы.**

**Сортировка Шелла  — алгоритм сортировки, являющийся усовершенствованным вариантом сортировки вставками. Принцип работы сортировки вставками:**

**Сортировка вставками — алгоритм сортировки, в котором элементы входной последовательности просматриваются по одному, и каждый новый поступивший элемент размещается в подходящее место среди ранее упорядоченных элементов.**

**Затраты памяти – O(n) всего, для хранения в массиве n элементов**

**Лучшее время – O(n), когда массив заранее отсортирован, и мы пробегаемся по его элементам**

**Худшее время - О(*n*2), когда пробегаемся по всем неотсортированным элементам массива и сравниваем их друг с другом**

**Среднее время - О(*n*2), когда пробегаемся по всем неотсортированным элементам массива и сравниваем их друг с другом, не отличается от худшего времени**

**Принцип работы сортировки Шелла:**

**Идея метода Шелла состоит в сравнении элементов, стоящих не только рядом, но и на определённом расстоянии друг от друга. Изначально это расстояние равно d или n/2, где n — общее число элементов. На первом шаге каждая группа включает в себя два элемента расположенных друг от друга на расстоянии n/2 или d; они сравниваются между собой, и, в случае необходимости, меняются местами. На последующих шагах также происходят проверка и обмен, но расстояние d сокращается на d/2, и количество групп, соответственно, уменьшается. Постепенно расстояние между элементами уменьшается, и на d=1 проход по массиву происходит в последний раз.**

1. **Оценка сложности.**

**Затраты памяти – O(n) всего, для хранения n элементов в массиве**

**Лучшее время - O(nlog2 *n*), глубина рекурсии log2 *n,* потому что каждый раз мы делим шаг на 2, происходит n сравнений (обменов) элементов, получается nlog2 *n***

**Худшее время - O(*n*2), все зависит от выбранного шага, если мы выберем шаг 1, то алгоритм полностью будет основываться на сортировке вставками, сложность будет такая же, как у вставок, это худший случай.**

**Среднее время - зависит от выбранных шагов, в настоящее время нет четкого представления о том, какое среднее время, потому что работа алгоритма будет зависеть от выбранного шага, но если изначально выбрать n/2, то будет близко к O(nlog2 *n*)**

1. **Зависимость времени от данных на linkedList**

|  |  |
| --- | --- |
| **data** | **time** |
| **100** | **70** |
| **200** | **320** |
| **300** | **750** |
| **400** | **1170** |
| **500** | **2000** |
| **600** | **3650** |
| **700** | **4800** |
| **800** | **6200** |
| **900** | **7490** |
| **1000** | **9470** |

**Зависимость времени от данных на массиве**

|  |  |
| --- | --- |
| **data** | **time** |
| **100** | **30** |
| **200** | **39** |
| **300** | **66** |
| **400** | **91** |
| **500** | **110** |
| **600** | **155** |
| **700** | **175** |
| **800** | **200** |
| **900** | **345** |
| **1000** | **332** |

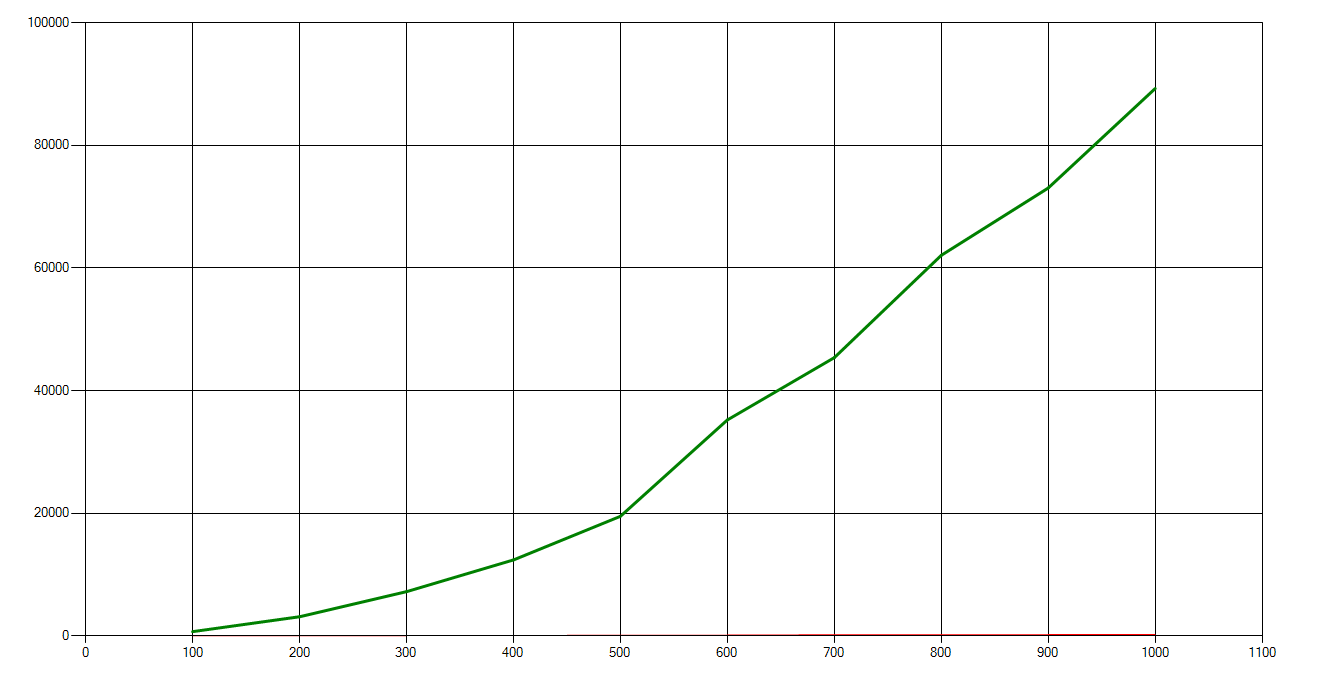
**Зависимость итераций от данных на массиве**

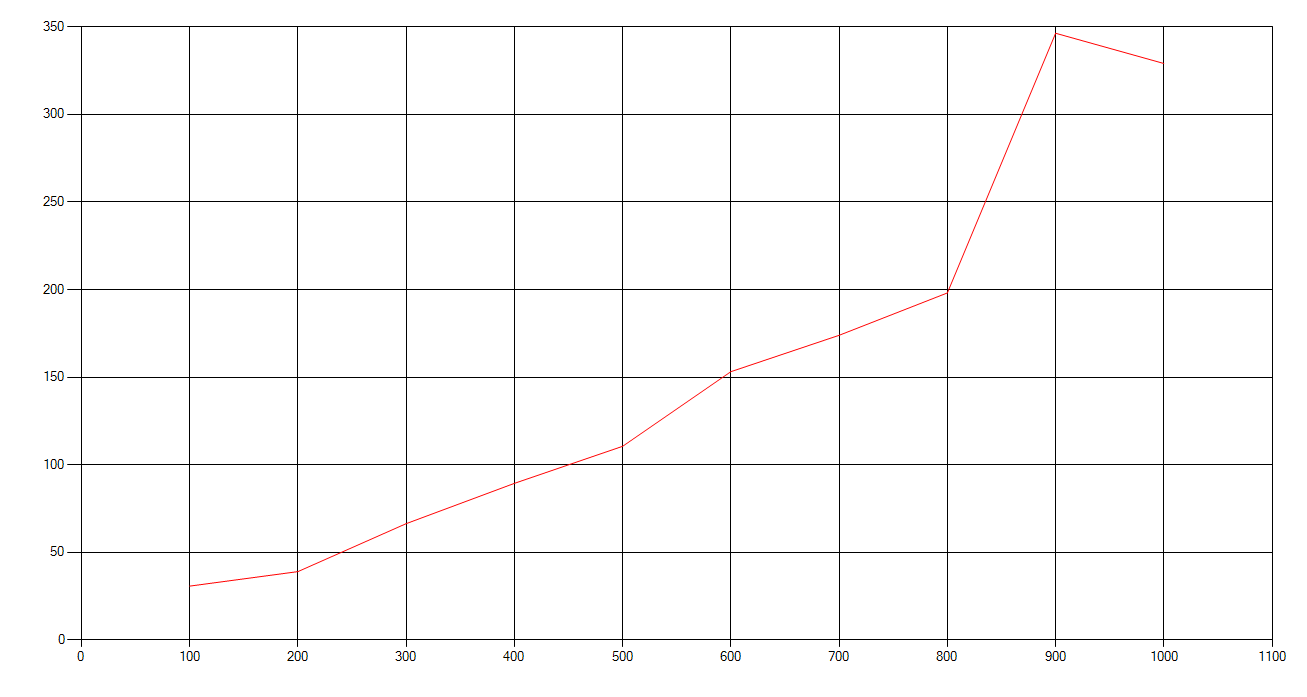
|  |  |
| --- | --- |
| **data** | **iterations** |
| **100** | **664** |
| **200** | **1492** |
| **300** | **2468** |
| **400** | **3400** |
| **500** | **4483** |
| **600** | **5537** |
| **700** | **6600** |
| **800** | **7710** |
| **900** | **8830** |
| **1000** | **9975** |

**Зависимость итераций от данных на linkedList**

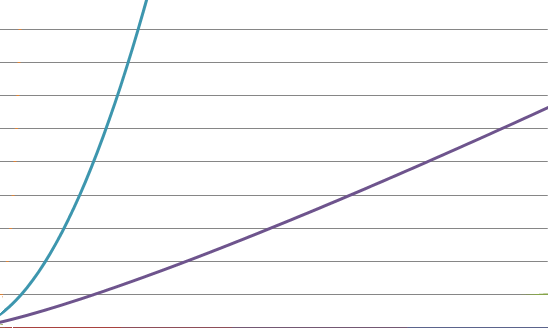
|  |  |
| --- | --- |
| **data** | **iterations** |
| **100** | **9800** |
| **200** | **38756** |
| **300** | **85374** |
| **400** | **15566** |
| **500** | **24010** |
| **600** | **34886** |
| **700** | **48772** |
| **800** | **63420** |
| **900** | **81000** |
| **1000** | **1000000** |

1. **Зависимость времени от количества данных на linkedList (зеленый) и массиве (красный), построено в Visual Studio.**





**Зависимость количества итераций от количества данных на linkedList (синий) и массиве (фиолетовый), построено в таблице Excel.**

****

1. **Выводы:**

**Плюсы алгоритма:**

* **отсутствие потребности в памяти под стек;**
* **отсутствие деградации при неудачных наборах данных — быстрая сортировка легко деградирует до O(n²), что хуже, чем худшее гарантированное время для сортировки Шелла.**
* **одна из быстрых сортировок**
* **устойчивый алгоритм, элементы с одинаковым значением не меняют своего взаимного расположения**
* **легко реализовать на массивах**

**Минусы алгоритма:**

* **сортировка Шелла во многих случаях медленнее, чем быстрая сортировка**
* **анализ сложности этого алгоритма – сложная математическая задача, у которой до сих пор нет полного решения, нет четкой средней сложности**

**Алгоритм применяется довольно часто, в связи с тем, что это одна из быстрых сортировок, после QuickSort и HeapSort.**

1. **Список использованной литературы:**

**Статья Википедии сортировка Шелла:** [**https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0\_%D0%A8%D0%B5%D0%BB%D0%BB%D0%B0**](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0_%D0%A8%D0%B5%D0%BB%D0%BB%D0%B0)

**Статья Википедии сортировка вставками:**

[**https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0\_%D0%B2%D1%81%D1%82%D0%B0%D0%B2%D0%BA%D0%B0%D0%BC%D0%B8**](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0_%D0%B2%D1%81%D1%82%D0%B0%D0%B2%D0%BA%D0%B0%D0%BC%D0%B8)