**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**Московский государственный технологический университет «СТАНКИН»**

**(ФГБОУ ВО МГТУ «СТАНКИН»)**

Кафедра «Управление и информатика в технических системах» Дисциплина: «Алгоритмы и структуры данных»

**Отчёт по лабораторным работам**

**Выполнила:** студентка Таган А.М.

группы ИДБ-23-13

**Проверил:** преподаватель доц. Евдокимов С.А.

МОСКВА 2024

**Содержание**

1. [Древовидная сортировка – что это и с чем едят. 3](#_Toc180251806)
2. [Описание алгоритма 5](#_Toc180251807)
3. [Графики времени сортировки 8](#_Toc180251808)

[Выводы 9](#_Toc180251809)

[Список использованных информационных источников 9](#_Toc180251810)

## **Древовидная** **сортировка – что это и с чем едят.**

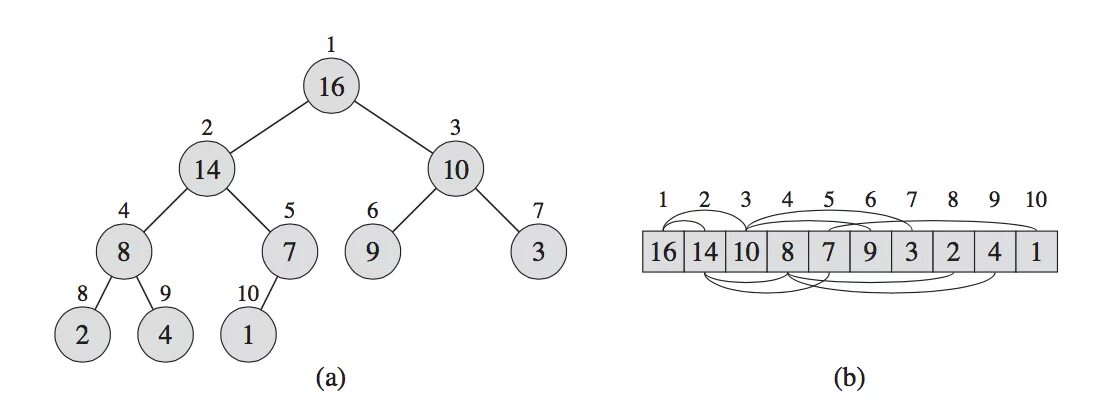
**Древовидная сортировка** — универсальный алгоритм сортировки, который **заключается в построении двоичного дерева поиска по ключам массива (списка)** с последующей сборкой результирующего массива путём обхода узлов построенного дерева в необходимом порядке следования ключей.

**Алгоритм древовидной сортировки**:

1. Взять входные элементы в виде массива.
2. Создать бинарное дерево поиска, вставив элементы данных из массива.
3. Выполнить обход дерева по порядку, чтобы получить элементы в отсортированном порядке.

Данная сортировка является оптимальной при получении данных путём непосредственного чтения из потока (например, файла, сокета или консоли).

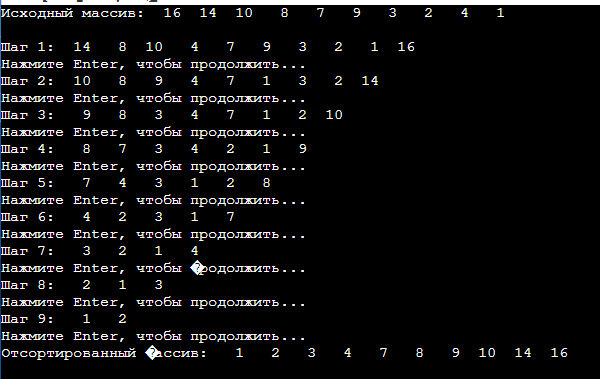
**Сложность древовидной сортировки**: процедура добавления объекта в бинарное дерево имеет среднюю алгоритмическую сложность порядка O(log(n)). Однако, сложность добавления объекта в разбалансированное дерево может достигать O(n), что может привести к общей сложности порядка O(n²).



Рис(1)

Как можно заметить по картинке выше, чтобы преобразовать обычный линейный массив в его древовидный вид надо знатно попотеть, используя особый алгоритм.

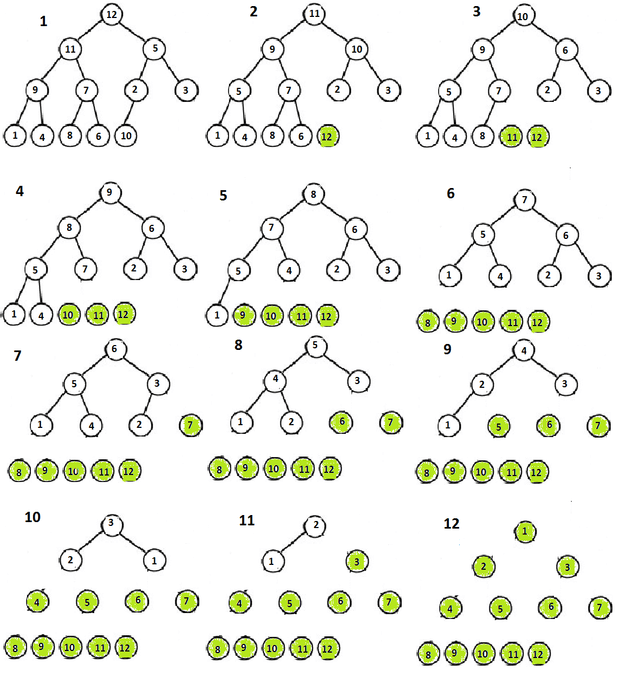
Ниже будет пример того как сортировка работает



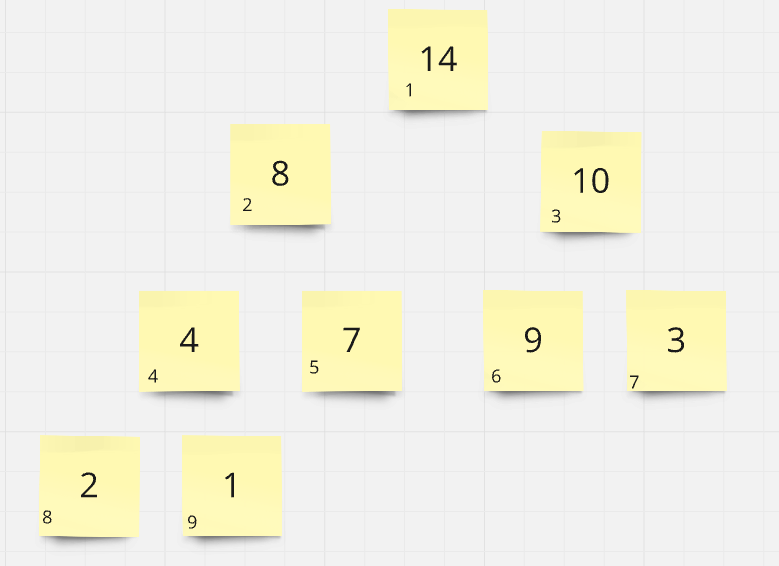
Рис(2)

Вот пример вывода алгоритма

## 2**. Описание алгоритма**

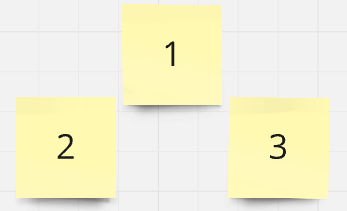


Рис(3)



Рис(4)

Как работает этот алгоритм?

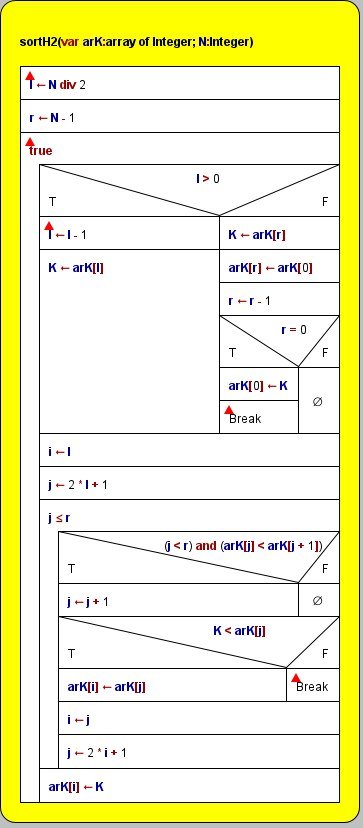
У нас есть массив который изображен линейно, чтобы сделать из него дерево, мы просто смотрим на индексы, например 1й элемент, после него идут 2 элемента, 3 и 4. 

Рис(5)

Вот как мы записываем древо, и в дальнейшем и прописываем остальные связи по такому же принципу.

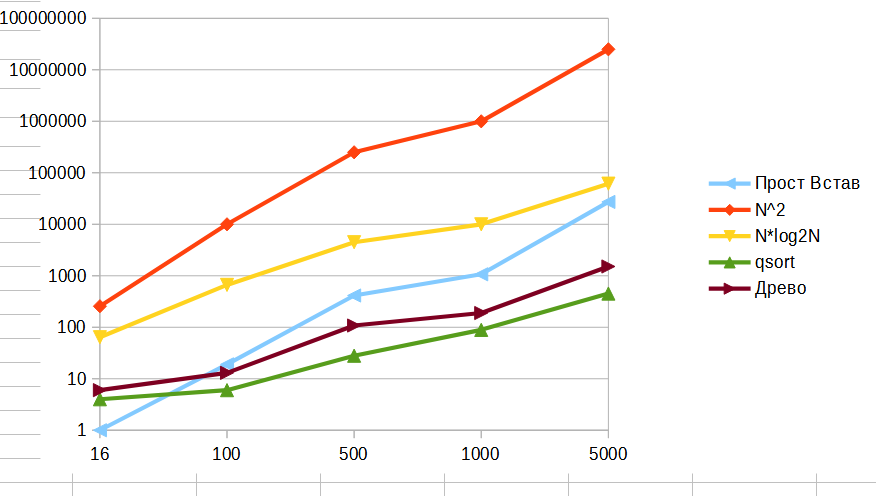
1 **Построение кучи:**   
   - Сначала исходный массив преобразуется в бинарную кучу. Это делается путем прохода по массиву снизу вверх, начиная с последнего нелистового узла.   
   - Для каждого узла выполняется операция, которая перестраивает поддерево, чтобы удовлетворять свойству кучи (корень больше/меньше всех своих потомков).  
  
2. **Извлечение элементов из кучи:**  
   - После построения кучи, самый большой (или самый маленький, в зависимости от того, какая сортировка нужна) элемент находится в корне.   
   - Корневой элемент удаляется (меняется местами с последним элементом кучи), после чего куча перестраивается, чтобы восстановить свойство кучи.  
   - Этот процесс повторяется, пока не будут удалены все элементы из кучи.  
  
**Пример по шагам (для массива [9, 12, 1, 7, 3, 10, 5, 8, 6, 2, 13, 14, 11, 4, 15, 0]):**  
  
1. **Построение кучи:**  
   - Начинаем с последнего нелистового узла (в нашем случае - 7). Выполняем "операцию построения дерева" для 7, 3, 10, 5, 8, 6, 2.   
   - Продолжаем для 12, 9, 1 (от корня к листьям).  
   - После этого мы получаем кучу, где 15 - корень (самый большой элемент).  
  
2. **Извлечение элементов:**  
   - Меняем местами 15 и 0 (последний элемент) - [0, 12, 1, 7, 3, 10, 5, 8, 6, 2, 13, 14, 11, 4, 15].  
   - Выполняем ""операцию построения дерева" для корня (0) - [15, 12, 1, 7, 3, 10, 5, 8, 6, 2, 13, 14, 11, 4, 0].  
   - Повторяем шаги:  
     - Меняем 15 и 4 - [4, 12, 1, 7, 3, 10, 5, 8, 6, 2, 13, 14, 11, 15, 0].  
     - для 4 - [15, 12, 1, 7, 3, 10, 5, 8, 6, 2, 13, 14, 11, 4, 0].  
     - И так далее, пока не останется один элемент.

Наибольшие элементы что встают в index = 0, мы закидываем в другой массив.



Рис(6) Блок схема

# 3. Графики времени сортировки



# Выводы

1. Сортировка древовидной сортировкой вставками имеет временную сложность n\*log2N
2. На малом числе элементов массива (N) в массиве:примерно меньших 50NN г. ) время сортировки простыми вставками лучше, чем у стандартной функции qsort;
3. Функция подобна функции qsort но всегда медленее

# Список использованных информационных источников

1. Кормен, Томас Х., Лейзерсон, Чарльз И., Ривест, Рональд Л., Штайн, Клиффорд. Алгоритмы: построение и анализ, 2-е издание. 2011 – 1296 стр.