Московский Авиационный Институт

(национальный исследовательский университет)

Факультет «Прикладной математики и информатики»

Кафедра 806

 Реферат на тему:

«Алгоритмы сортировки»

  Выполнил: студент 1 курса

Каширин К.Д.

Преподаватель: Трубченко Н.М.

Москва 2020

**Содержание**

Введение ……………………………………………...…………………………………………….. 3

История ….…………….…………………………………...…………...............................................4

Классификация алгоритмов сортировок…...………………….…..…………………….…......…..7

Сортировка пузырьком………………...………………………………………………......…..........9

Сортировка выбором…………………………………………………………......…….....……..... 11

Сортировка вставками………………..………………….………………………..….……......………………....12

Сортировка Шелла………………………...…….....……………………...............….……........... 14

Быстрая сортировка…………………………………………………….…..........……..…………..16

Сортировка слиянием……………………………………………………………………………....18

Пирамидальная сортировка………………………………………………………………………..20

Заключение……………………………………………………………………………………….... 22

Список использованной литературы………………………………………………………….…..23

# **Введение**

Алгоритмы сортировки обрабатывают массивы элементов любого типа. Такая задача подразумевает упорядочивание элементов массива в определенном порядке. Обычно по возрастанию и убыванию данных. Упорядочить набор данных означает переставить элементы в определенном порядке так, чтобы шло возрастание или убывание с каждым шагом.

Известные алгоритмы сортировки данных, расположенных в оперативной памяти, чрезвычайно разнообразны. Их анализ очень полезен с точки зрения обучения, так как в них используются практически все универсальные приемы конструирования алгоритмов любой сложности.

Алгоритм сортировки - это алгоритм, который помогает упорядочить набор данных в таблице в определенную последовательность. Обычно, массивы сортируют по убыванию и возрастанию.

В связи с разнообразием задач на сортировку данных таблиц, существует много разных метод сортировки, которые целесообразно использовать в различных ситуациях, в целях экономии средств компьютера и времени пользователя.

**История**

Первые прототипы современных методов сортировки появились уже в XIX веке. К 1890 году для ускорения обработки данных [переписи населения](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D0%B5%D0%BF%D0%B8%D1%81%D1%8C_%D0%BD%D0%B0%D1%81%D0%B5%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F) в [США](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%B5%D0%B4%D0%B8%D0%BD%D1%91%D0%BD%D0%BD%D1%8B%D0%B5_%D0%A8%D1%82%D0%B0%D1%82%D1%8B_%D0%90%D0%BC%D0%B5%D1%80%D0%B8%D0%BA%D0%B8) американец [Герман Холлерит](https://ru.wikipedia.org/wiki/%D0%93%D0%B5%D1%80%D0%BC%D0%B0%D0%BD_%D0%A5%D0%BE%D0%BB%D0%BB%D0%B5%D1%80%D0%B8%D1%82) создал первый статистический [табулятор](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%B1%D1%83%D0%BB%D1%8F%D1%82%D0%BE%D1%80) — электромеханическую машину, предназначенную для автоматической обработки информации, записанной на [перфокартах](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%84%D0%BE%D0%BA%D0%B0%D1%80%D1%82%D0%B0). У машины Холлерита имелся специальный «сортировальный ящик» из 26 внутренних отделений. При работе с машиной от оператора требовалось вставить перфокарту и опустить рукоятку. Благодаря пробитым на перфокарте отверстиям замыкалась определённая [электрическая цепь](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%86%D0%B5%D0%BF%D1%8C), и на единицу увеличивалось показание связанного с ней циферблата. Одновременно с этим открывалась одна из 26 крышек сортировального ящика, и в соответствующее отделение перемещалась перфокарта, после чего крышка закрывалась. Данная машина позволила обрабатывать около 50 карт в минуту, что ускорило обработку данных в 3 раза. К переписи населения 1900 года Холлерит усовершенствовал машину, автоматизировав подачу карт. Работа сортировальной машины Холлерита основывалась на методах [поразрядной сортировки](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D0%B0%D0%B7%D1%80%D1%8F%D0%B4%D0%BD%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0). В патенте на машину обозначена сортировка «по отдельности для каждого столбца», но не определён порядок. В другой аналогичной машине, запатентованной в 1894 году Джоном Гором, упоминается сортировка со столбца десятков. Метод сортировки, начиная со столбца единиц, впервые появляется в литературе в конце 1930-х годов. К этому времени сортировальные машины уже позволяли обрабатывать до 400 карт в минуту.

В дальнейшем история алгоритмов оказалась связана с развитием [электронно-вычислительных машин](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%BE-%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%88%D0%B8%D0%BD%D0%B0). По некоторым источникам, именно программа сортировки стала первой программой для вычислительных машин. Некоторые конструкторы ЭВМ, в частности разработчики [EDVAC](https://ru.wikipedia.org/wiki/EDVAC), называли задачу сортировки данных наиболее характерной нечисловой задачей для вычислительных машин. В 1945 году [Джон фон Нейман](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%B9%D0%BC%D0%B0%D0%BD,_%D0%94%D0%B6%D0%BE%D0%BD_%D1%84%D0%BE%D0%BD) для тестирования ряда команд для EDVAC разработал программы [сортировки методом слияния](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_%D1%81%D0%BB%D0%B8%D1%8F%D0%BD%D0%B8%D0%B5%D0%BC). В том же году немецкий инженер [Конрад Цузе](https://ru.wikipedia.org/wiki/%D0%A6%D1%83%D0%B7%D0%B5,_%D0%9A%D0%BE%D0%BD%D1%80%D0%B0%D0%B4) разработал программу для [сортировки методом простой вставки](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). К этому времени уже появились быстрые специализированные сортировальные машины, в сопоставлении с которыми и оценивалась эффективность разрабатываемых ЭВМ. Первым опубликованным обсуждением сортировки с помощью вычислительных машин стала лекция [Джона Мокли](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%BA%D0%BB%D0%B8,_%D0%94%D0%B6%D0%BE%D0%BD), прочитанная им в 1946 году. Мокли показал, что сортировка может быть полезной также и для численных расчетов, описал методы сортировки [простой вставки и бинарных вставок](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), а также [поразрядную сортировку](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D0%B0%D0%B7%D1%80%D1%8F%D0%B4%D0%BD%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) с частичными проходами. Позже организованная им совместно с инженером [Джоном Эккертом](https://ru.wikipedia.org/wiki/%D0%AD%D0%BA%D0%BA%D0%B5%D1%80%D1%82,_%D0%94%D0%B6%D0%BE%D0%BD_%D0%9F%D1%80%D0%B5%D1%81%D0%BF%D0%B5%D1%80) компания «[Eckert–Mauchly Computer Corporation](https://ru.wikipedia.org/wiki/Eckert%E2%80%93Mauchly_Computer_Corporation)» выпустила некоторые из самых ранних электронных вычислительных машин [BINAC](https://ru.wikipedia.org/wiki/BINAC) и [UNIVAC](https://ru.wikipedia.org/wiki/UNIVAC). Наряду с отмеченными алгоритмами [внутренней сортировки](https://ru.wikipedia.org/wiki/%D0%92%D0%BD%D1%83%D1%82%D1%80%D0%B5%D0%BD%D0%BD%D1%8F%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0), появлялись алгоритмы [внешней сортировки](https://ru.wikipedia.org/wiki/%D0%92%D0%BD%D0%B5%D1%88%D0%BD%D1%8F%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0), развитию которых способствовал ограниченный объём памяти первых вычислительных машин. В частности, были предложены методы сбалансированной двухпутевой поразрядной сортировки и сбалансированного двухпутевого слияния. К 1952 году на практике уже применялись многие методы внутренней сортировки, но теория была развита сравнительно слабо. В октябре 1952 года Даниэль Гольденберг привёл пять методов сортировки с анализом наилучшего и наихудшего случаев для каждого из них. В 1954 году Гарольд Сьюворд развил идеи Гольденберга, а также проанализировал методы внешней сортировки. Говард Демут в 1956 году рассмотрел три абстрактные модели задачи сортировки: с использованием циклической памяти, линейной памяти и памяти с произвольным доступом. Для каждой из этих задач автор предложил оптимальные или почти оптимальные методы сортировки, что помогло связать теорию с практикой. Из-за малого числа людей, связанных с вычислительной техникой, эти доклады не появлялись в «открытой литературе». Первой большой обзорной статьёй о сортировке, появившейся в печати в 1955 году, стала работа Дж. Хоскена, в которой он описал всё имевшееся на тот момент оборудование специального назначения и методы сортировки для ЭВМ, основываясь на брошюрах фирм-изготовителей. В 1956 году Э. Френд в своей работе проанализировал математические свойства большого числа алгоритмов [внутренней](https://ru.wikipedia.org/wiki/%D0%92%D0%BD%D1%83%D1%82%D1%80%D0%B5%D0%BD%D0%BD%D1%8F%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) и [внешней сортировки](https://ru.wikipedia.org/wiki/%D0%92%D0%BD%D0%B5%D1%88%D0%BD%D1%8F%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0), предложив некоторые новые методы.

После этого было предложено множество различных алгоритмов сортировки: например, вычисление адреса в 1956 году; слияние с вставкой, [обменная поразрядная сортировка](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%80%D0%B0%D0%B7%D1%80%D1%8F%D0%B4%D0%BD%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0), каскадное слияние и [метод 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) в 1959 году, [многофазное слияние](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_%D1%81%D0%BB%D0%B8%D1%8F%D0%BD%D0%B8%D0%B5%D0%BC) и вставки в дерево в 1960 году, осциллирующая сортировка и [быстрая сортировка Хоара](https://ru.wikipedia.org/wiki/%D0%91%D1%8B%D1%81%D1%82%D1%80%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) в 1962 году, [пирамидальная сортировка Уильямса](https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%80%D0%B0%D0%BC%D0%B8%D0%B4%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) и обменная сортировка со слиянием [Бэтчера](https://ru.wikipedia.org/wiki/%D0%91%D1%8D%D1%82%D1%87%D0%B5%D1%80,_%D0%9A%D0%B5%D0%BD%D0%BD%D0%B5%D1%82_%D0%AD%D0%B4%D0%B2%D0%B0%D1%80%D0%B4) в 1964 году. В конце 60-х годов произошло и интенсивное развитие теории сортировки. Появившиеся позже алгоритмы во многом являлись вариациями уже известных методов. Получили распространение адаптивные методы сортировки, ориентированные на более быстрое выполнение в случаях, когда входная последовательность удовлетворяет заранее установленным критериям. В связи с разнообразием задач на сортировку данных таблиц, существует много разных метод сортировки, которые целесообразно использовать в различных ситуациях, в целях экономии средств компьютера и времени пользователя.

**Классификация алгоритмов сортировки**

Алгоритмы классифицируются по устойчивости, естественностью поведения, использование операции сравнения:

Устойчивость (stability) — устойчивая сортировка не меняет взаимного расположения равных элементов.

Естественность поведения — эффективность метода при обработке уже упорядоченных, или частично упорядоченных данных. Алгоритм ведёт себя естественно, если учитывает эту характеристику входной последовательности и работает лучше.

Использование операции сравнения. Алгоритмы, использующие для сортировки сравнение элементов между собой, называются основанными на сравнениях. Минимальная трудоемкость худшего случая для этих алгоритмов составляет , но они отличаются гибкостью применения. Для специальных случаев (типов данных) существуют более эффективные алгоритмы.

Ещё одним важным свойством алгоритма является его сфера применения. Здесь основных типов упорядочения два:

1. Внутренняя сортировка оперирует с массивами, целиком помещающимися в оперативной памяти с произвольным доступом к любой ячейке. Данные обычно упорядочиваются на том же месте, без дополнительных затрат.

В современных архитектурах персональных компьютеров широко применяется подкачка и кэширование памяти. Алгоритм сортировки должен хорошо сочетаться с применяемыми алгоритмами кэширования и подкачки.

2. Внешняя сортировка оперирует с запоминающими устройствами большого объёма, но с доступом не произвольным, а последовательным (упорядочение файлов), т. е. в данный момент мы 'видим' только один элемент, а затраты на перемотку по сравнению с памятью неоправданно велики. Это накладывает некоторые дополнительные ограничения на алгоритм и приводит к специальным методам упорядочения, обычно использующим дополнительное дисковое пространство. Кроме того, доступ к данным на носителе производится намного медленнее, чем операции с оперативной памятью.

Доступ к носителю осуществляется последовательным образом: в каждый момент времени можно считать или записать только элемент, следующий за текущим.

Объём данных не позволяет им разместиться в ОЗУ.

Также алгоритмы классифицируются по:

- потребности в дополнительной памяти или её отсутствии

- потребности в знаниях о структуре данных, выходящих за рамки операции сравнения, или отсутствии таковой

Алгоритмы устойчивой сортировки

1. Сортировка пузырьком (англ. Bubble sort ) — сложность алгоритма: O(n2); для каждой пары индексов производится обмен, если элементы расположены не по порядку.

2. Сортировка перемешиванием (Шейкерная, Cocktail sort, bidirectional bubble sort) — Сложность алгоритма: O(n2)

3. Сортировка вставками (Insertion sort) — Сложность алгоритма: O(n2); определяем где текущий элемент должен находиться в упорядоченном списке и вставляем его туда

4. Сортировка подсчётом (Counting sort) — Сложность алгоритма: O(n+k); требуется O(n+k) дополнительной памяти (рассмотрено 3 варианта)

5. Сортировка слиянием (Merge sort) — Сложность алгоритма: O(n log n); требуется O(n) дополнительной памяти; выстраиваем первую и вторую половину списка отдельно, а затем — сливаем упорядоченные списки

Алгоритмы неустойчивой сортировки

1.Сортировка выбором (Selection sort) — Сложность алгоритма: O(n2); поиск наименьшего или наибольшего элемента и помещения его в начало или конец упорядоченного списка

2.Сортировка Шелла (Shell sort) — Сложность алгоритма: O(n log2 n); попытка улучшить сортировку вставками

3.Сортировка расчёской (Comb sort) — Сложность алгоритма: O(n log n)

4.Пирамидальная сортировка (Сортировка кучи, Heapsort) — Сложность алгоритма: O(n log n); превращаем список в кучу, берём наибольший элемент и добавляем его в конец списка

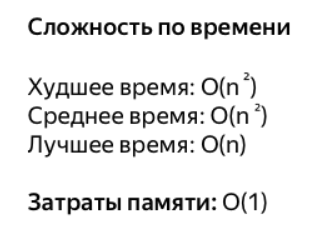
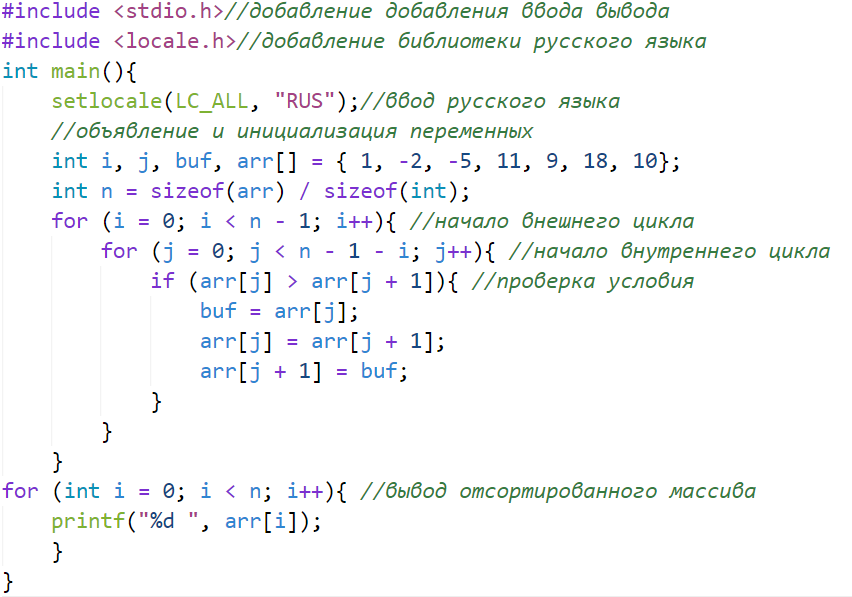
5.Плавная сортировка (Smoothsort) — Сложность алгоритма: O(n log n)

6.Быстрая сортировка (Quicksort), в варианте с минимальными затратами памяти — Сложность алгоритма: O(n log n) — среднее время, O(n2) — худший случай; широко известен как быстрейший из известных для упорядочения больших случайных списков; с разбиением исходного набора данных на две половины так, что любой элемент первой половины упорядочен относительно любого элемента второй половины; затем алгоритм применяется рекурсивно к каждой половине. При использовании O(n) дополнительной памяти, можно сделать сортировку устойчивой.

# **Сортировка пузырьком**

Метод пузырька (сортировка пузырьком) является, пожалуй, наиболее распространенным алгоритмом сортировки данных в массиве.

Алгоритм состоит из повторяющихся проходов по сортируемому массиву. За каждый проход элементы последовательно сравниваются попарно и, если порядок в паре неверный, выполняется обмен элементов. Проходы по массиву повторяются N-1 раз или до тех пор, пока на очередном проходе не окажется, что обмены больше не нужны, что означает — массив отсортирован. При каждом проходе алгоритма по внутреннему циклу, очередной наибольший элемент массива ставится на своё место в конце массива рядом с предыдущим «наибольшим элементом», а наименьший элемент перемещается на одну позицию к началу массива («всплывает» до нужной позиции, как пузырёк в воде — отсюда и название алгоритма).

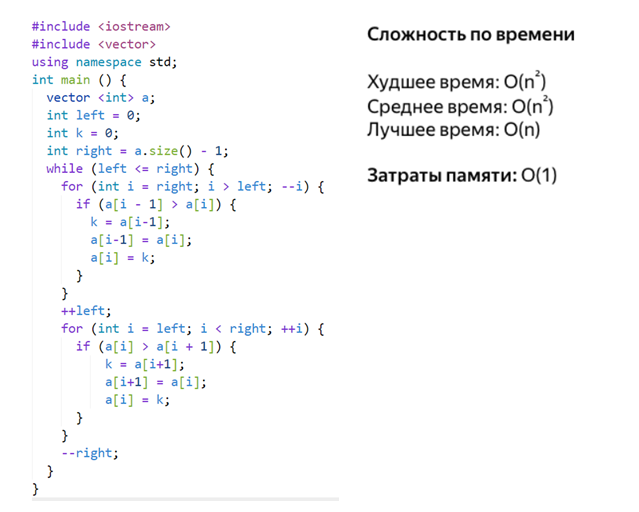
На языке программирования С код сортировки пузырьком выглядит так: 

Так как на прохождение всего массива требуется очень много времени, то можно заключить, что алгоритм пузырька очень медленен и малоэффективен. Тем не менее, он все же имеет плюс: его можно легко улучшать за счет его простоты.

Рассмотрим улучшения, которые можно привнести в этот алгоритм.

Во-первых, можно запоминать производился ли на данном проходе как-либо обмен, и если нет, то алгоритм заканчивает работу. Это позволит избежать излишнего прохода по массиву, когда и так понятно, что он уже отсортирован. Это улучшение можно так же модернизировать, запоминая не только событие обмена, но и индекс последнего обмена n. Все пары соседних элементов с индексами, меньшими n, уже расположены в нужном порядке. Дальнейшие проходы можно заканчивать на индексе n, вместо того чтобы двигаться до установленной заранее верхней границы i.

Еще одним качественным улучшением является модернизация алгоритма до Шейкер-сортировки. Такая сортировка отличается тем, что в ней проходы по массиву с каждым шагом меняют свое направление.

Код такой сортировки на С++ выглядит так: 

Такие улучшения хоть и уменьшают время, требуемое на отсортировку массива, но не уменьшают количество требуемых обменов.

На практике метод пузырька почти не применяется.

# **Сортировка выбором**

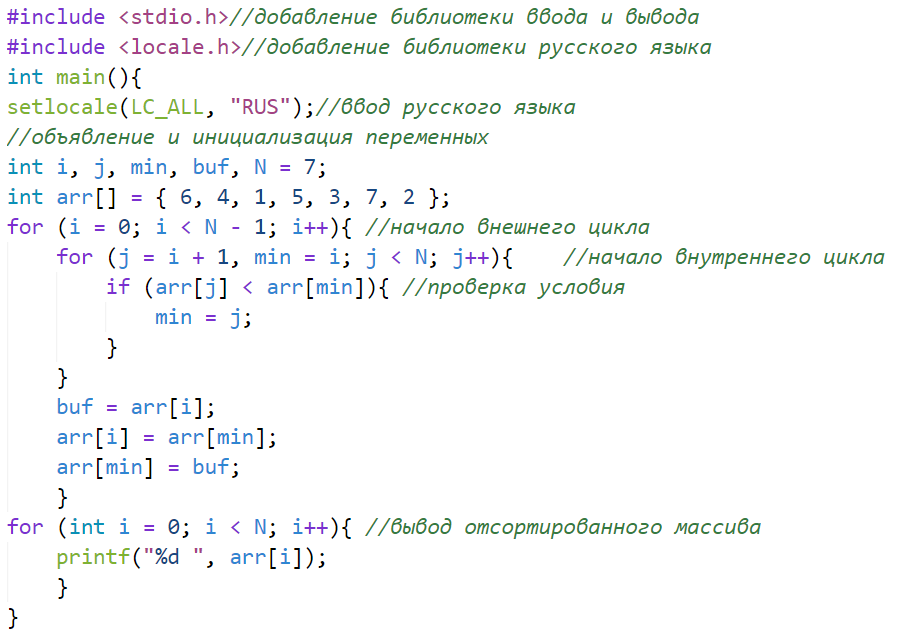
Идея сортировки этого алгоритма состоит в том, что отсортированная последовательность формируется путем добавления к ней элементов в правильном порядке.

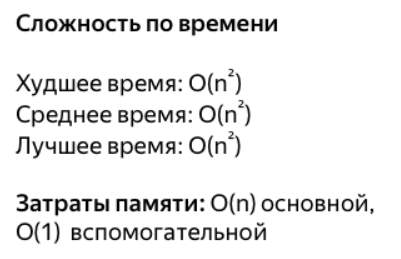
Алгоритм основан на выборе наименьшего элемента из неотсортированной части и его переноса в конец отсортированной. Сначала программа пробегает по массиву, выбирая наименьший элемент. Как только такой найден, он переносится на первое место в массиве, а элемент стоящий там, съезжает на свободное место. С этого момента, первый элемент, он же наименьший, будет началом отсортированной части массива. Вторым шагом, мы ищем наименьший элемент среди неотсортированной части и, найдя его, ставим его в конец отсортированной части. И так продолжаем циклически, пока весь массив не окажется отсортированным.

Для нахождения наименьшего элемента из массива рассматриваемых алгоритм совершает n сравнений. Тогда, как число обменов всегда будет меньше числа сравнений. То есть время, затраченное на сортировку, растет квадратично, относительно количества элементов.

Алгоритм сортировки выбором не использует дополнительной памяти, то есть все операции происходят "на месте", без дополнительных массивов.

Этот метод нельзя назвать устойчивым, так как при сортировке последовательности, состоящей из 3 элементов, которые имеют по 2 поля, например: 1-a, 2-a, 2-b, алгоритм отсортирует последовательность в порядке: 1-a, 2-b, 2-a, что в данной ситуации является неверным.

 Код сортировки выбором на языке С:



# **Сортировка вставками**

Сортировка вставками чем-то напоминает вышеизложенные методы. Главным ее отличием является то, что массив, с которым работает алгоритм, предварительно отсортирован наполовину.

Главной идеей такой сортировки является сравнение первого элемента из неотсортированной части с последним элементом отсортированной, и если он больше, то мы заносим его в отсортированную последовательность последним элементом, если же он оказывается меньше, то мы сравниваем этот элемент с предпоследним элементом отсортированной последовательности. И так продолжается в цикле, пока весь массив не будет отсортирован.



Рисунок 1. Пример работы алгоритма сортировки вставками

При сортировке простыми вставками не задействуется дополнительная память. Алгоритм является очень устойчивым. Также к хорошим показателям такого метода можно отнести скорость сортировки: почти отсортированный массив будет упорядочен за n\*n времени.

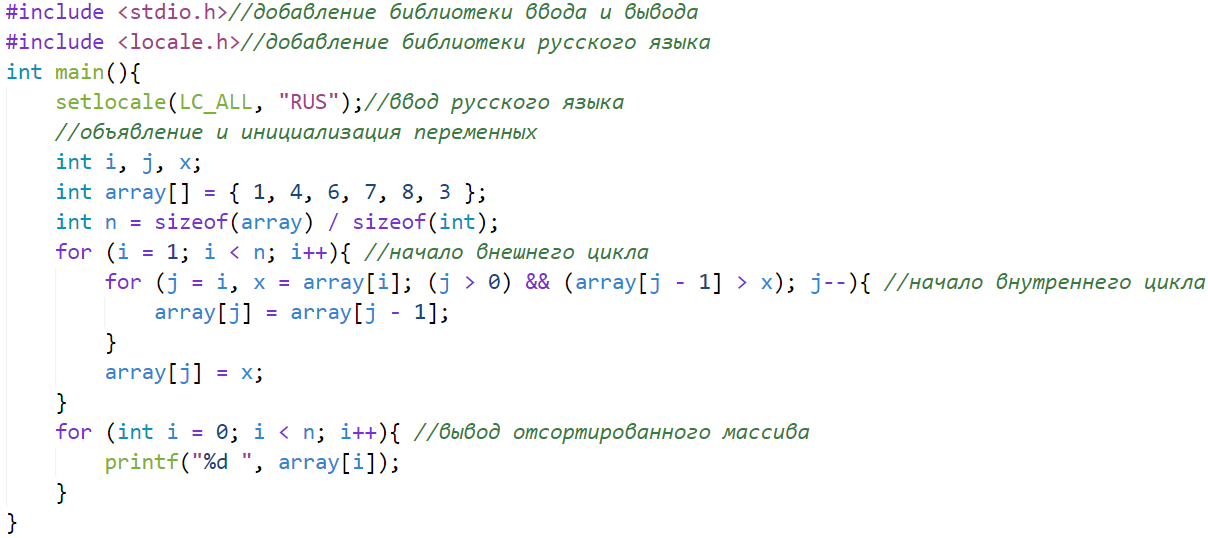
Алгоритм сортировки простыми вставками можно улучшить, добавлением в начало массива элемента, заведомо меньше, чем все остальные. Тогда при j=0 будет заведомо верно условие a [0] <= x. Цикл остановится на нулевом элементе, что и было целью условия j>=0.

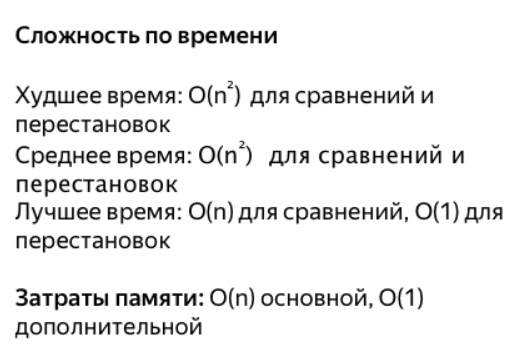
Так сортировка будет протекать правильно, и за счет уменьшения одного условия, увеличится скорость выполнения алгоритма. А с учетом того, что алгоритм выполнялся за n\*n это становится хорошим преимуществом.

Однако в таком виде, мы теряем первый элемент массива, вместо которого мы ставили минимум. Для окончательной сортировки, нужно вставить его на место, а потом на корректное место в отсортированную последовательность.

Такой минимальный элемент называют сторожевым элементом, а сортировку - сортировкой вставками со сторожевым элементом.

Код сортировки вставками на языке С:





# **Сортировка Шелла**

Метод Шелла является модификацией алгоритма сортировки вставками. Рассмотрим работу алгоритма на примере сортировки массива

а [0] - > а [15].

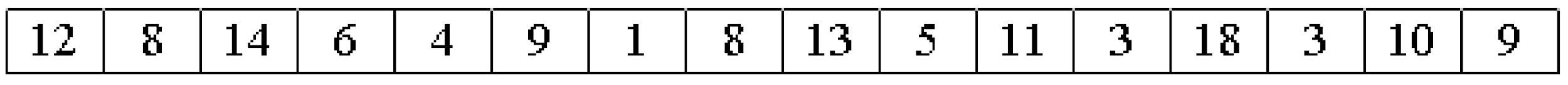


Рисунок 3. Метод Шелла (1)

Сортируем вставками 8 групп из двух элементов (Рисунок 4)

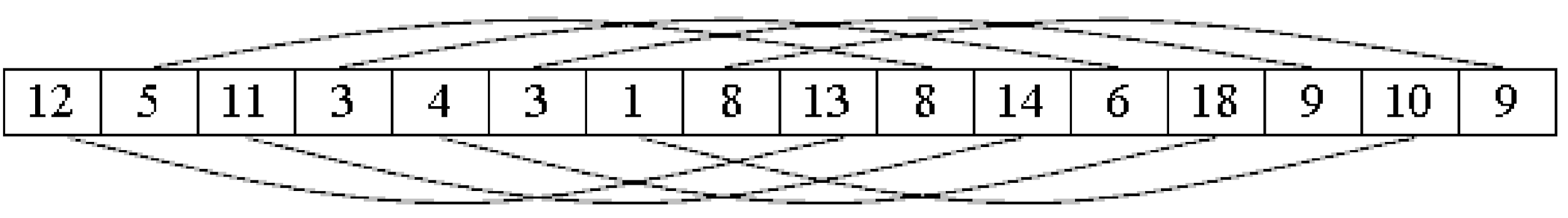


Рисунок 4. Метод Шелла (2)

. Далее сортируем группы из четырех элементов (Рисунок 5)

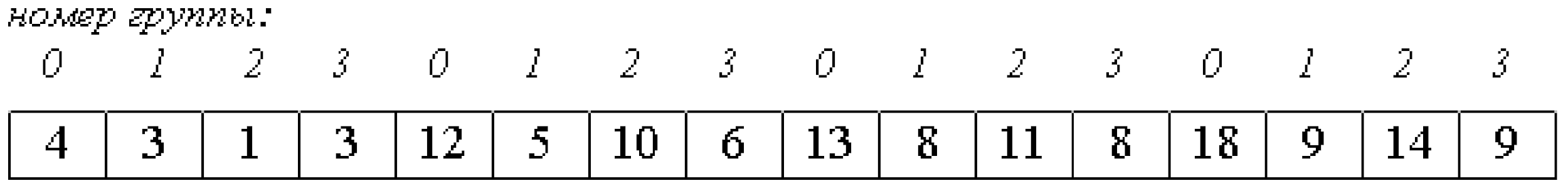


Рисунок 5. Метод Шелла (3)

. Теперь сортируем две группы по восемь элементов (Рисунок 6)

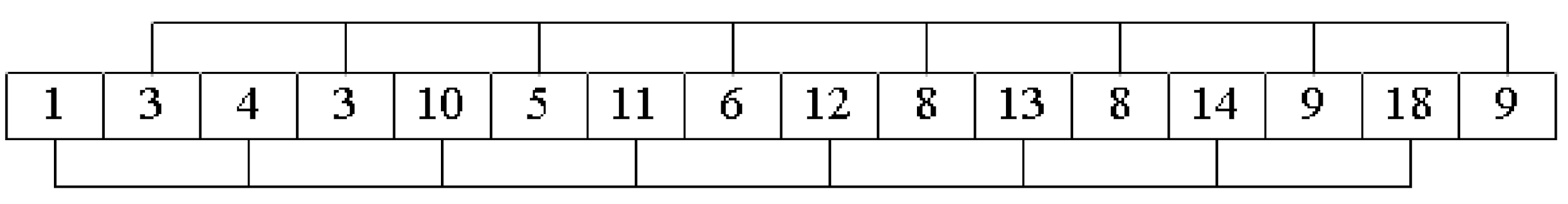


Рисунок 6. Метод Шелла (4)

. Сортируем вставками все элементы.

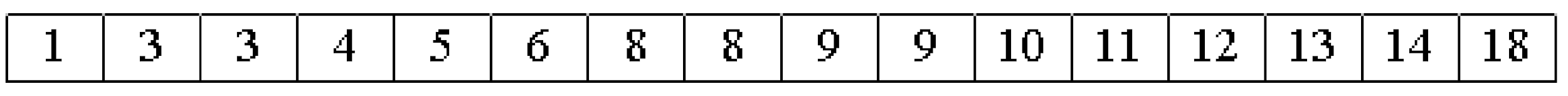


Рисунок 7. Метод Шелла (5)

В данном алгоритме первые три предварительные сортировки необходимы, чтобы подвинуть элементы к своим правильным позициям. Тогда как последняя сортировка окончательно загоняет их на свои места.

Доказано многими исследованиями, что такая модифицированная сортировка значительно ускоряет процесс упорядочения массива.

Единственной характеристикой сортировки Шелла является приращение - расстояние между сортируемыми элементами, в зависимости от прохода. В конце приращение всегда равно единице - метод завершается обычной сортировкой вставками, но именно последовательность приращений определяет рост эффективности.

Р. Седжвик предложил выбор порядка сортировки методом Шелла.

Его последовательность имеет вид:

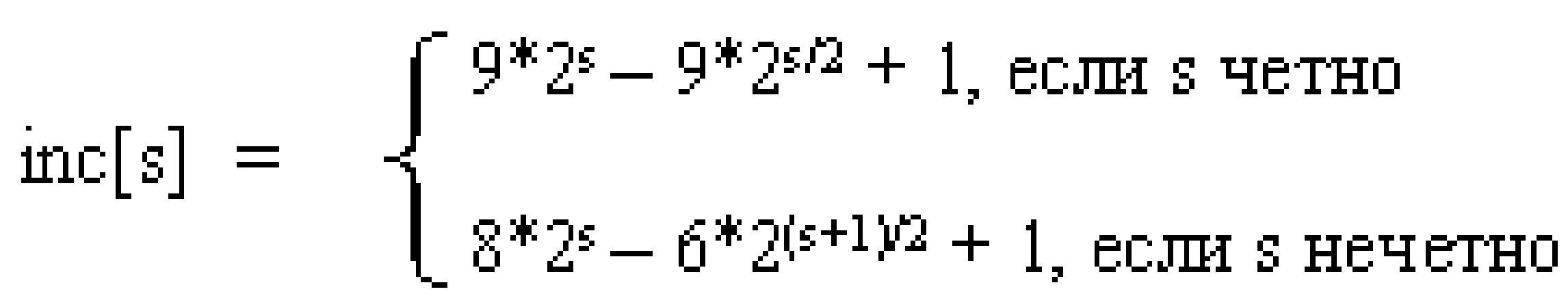
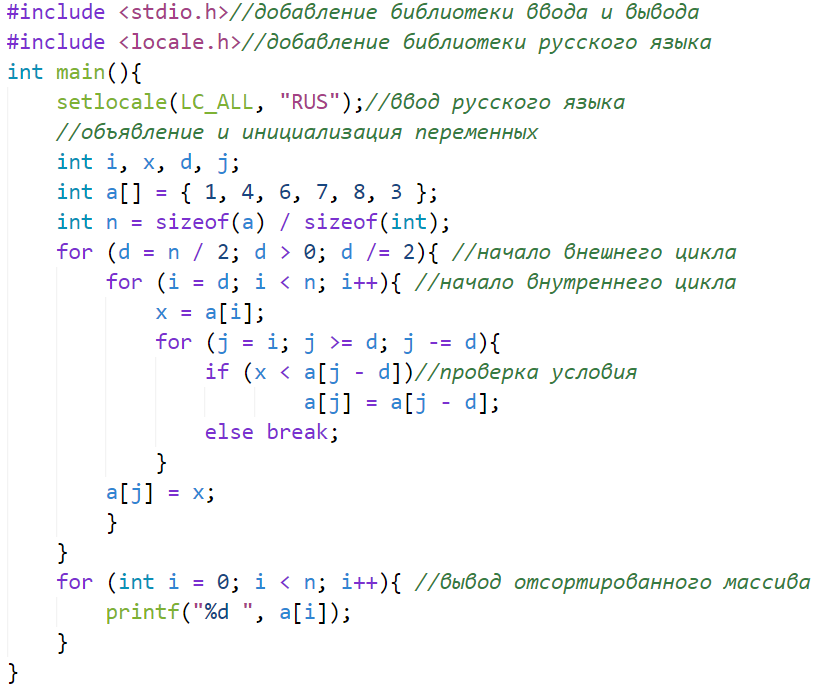


Рисунок 8. Выбор сортировки

При использовании таких приращений среднее количество операций становится О (n7/6), что является меньшим, чем при обычной сортировки вставками.

Реализация на языке программирования С:



# **Быстрая сортировка**

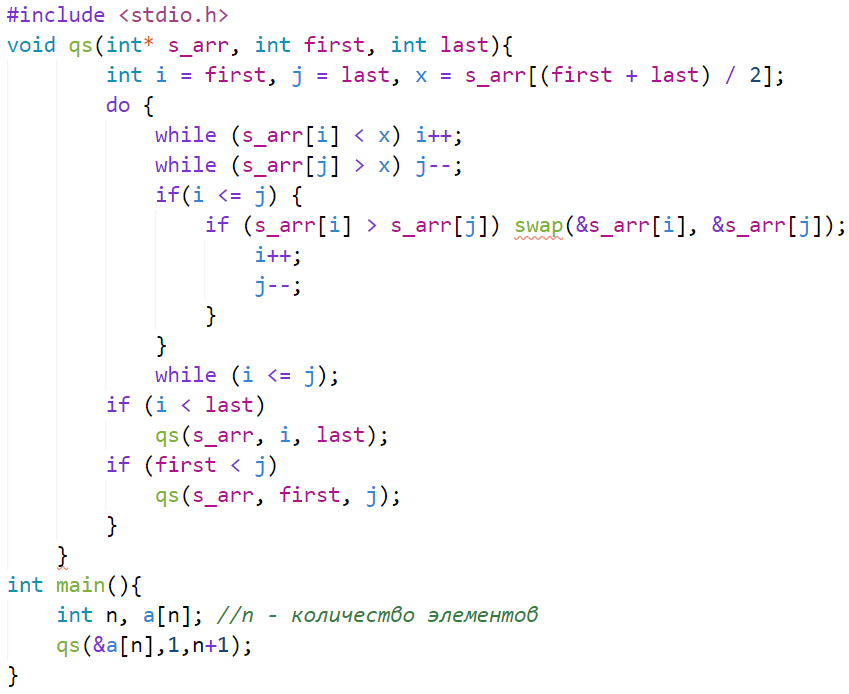
Быстрая сортировка является одним из самых старых методов сортировки, при этом она одна из наиболее используемых и наиболее эффективных методов сортировки. Быструю сортировку изобрели в 1960 году для машинного перевода: тогда словари хранились на магнитных лентах, а сортировка слов обрабатываемого текста позволяла получить переводы за один прогон ленты, без перемотки назад.

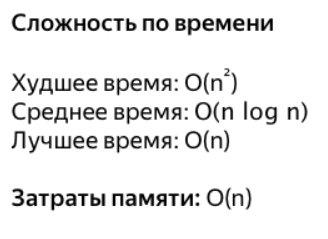
Быстрая сортировка или Quiсk Sоrt основана на методе "разделяй и властвуй". Этот алгоритм состоит из трёх шагов. Сначала из массива нужно выбрать один элемент — его обычно называют опорным. Затем другие элементы в массиве перераспределяют так, чтобы элементы меньше опорного оказались до него, а большие или равные — после. А дальше рекурсивно применяют первые два шага к подмассивам справа и слева от опорного значения. Такой алгоритм допускает написание рекурсии.

Метод считается неустойчивым. При частично упорядоченном массиве, его удается разделить на более-менее равные части. Сортировка использует дополнительную память из-за наличия рекурсии.

Этот алгоритм состоит из трёх шагов. Сначала из массива нужно выбрать один элемент — его обычно называют опорным. Затем другие элементы в массиве перераспределяют так, чтобы элементы меньше опорного оказались до него, а большие или равные — после. А дальше рекурсивно применяют первые два шага к подмассивам справа и слева от опорного значения.

Код реализации на С:





**Сортировка слиянием**

Сортировка слиянием — [алгоритм сортировки](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B8), который упорядочивает [списки](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_(%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0)) (или другие [структуры данных](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85), доступ к [элементам](https://ru.wikipedia.org/wiki/%D0%9C%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%BE#%D0%AD%D0%BB%D0%B5%D0%BC%D0%B5%D0%BD%D1%82_%D0%BC%D0%BD%D0%BE%D0%B6%D0%B5%D1%81%D1%82%D0%B2%D0%B0) которых можно получать только [последовательно](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%81%D0%BB%D0%B5%D0%B4%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF), например — [потоки](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D1%82%D0%BE%D0%BA_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85)) в определённом порядке. Эта сортировка — хороший пример использования принципа «[разделяй и властвуй](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D0%B4%D0%B5%D0%BB%D1%8F%D0%B9_%D0%B8_%D0%B2%D0%BB%D0%B0%D1%81%D1%82%D0%B2%D1%83%D0%B9_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5))». Сначала задача разбивается на несколько подзадач меньшего размера. Затем эти задачи решаются с помощью [рекурсивного вызова](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BA%D1%83%D1%80%D1%81%D0%B8%D1%8F) или непосредственно, если их размер достаточно мал. Наконец, их [решения](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_%D0%B7%D0%B0%D0%B4%D0%B0%D1%87) комбинируются, и получается решение исходной задачи.

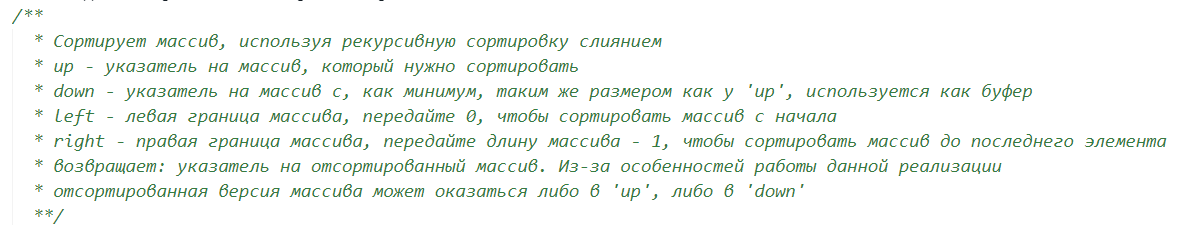
Для решения задачи сортировки эти три этапа выглядят так:

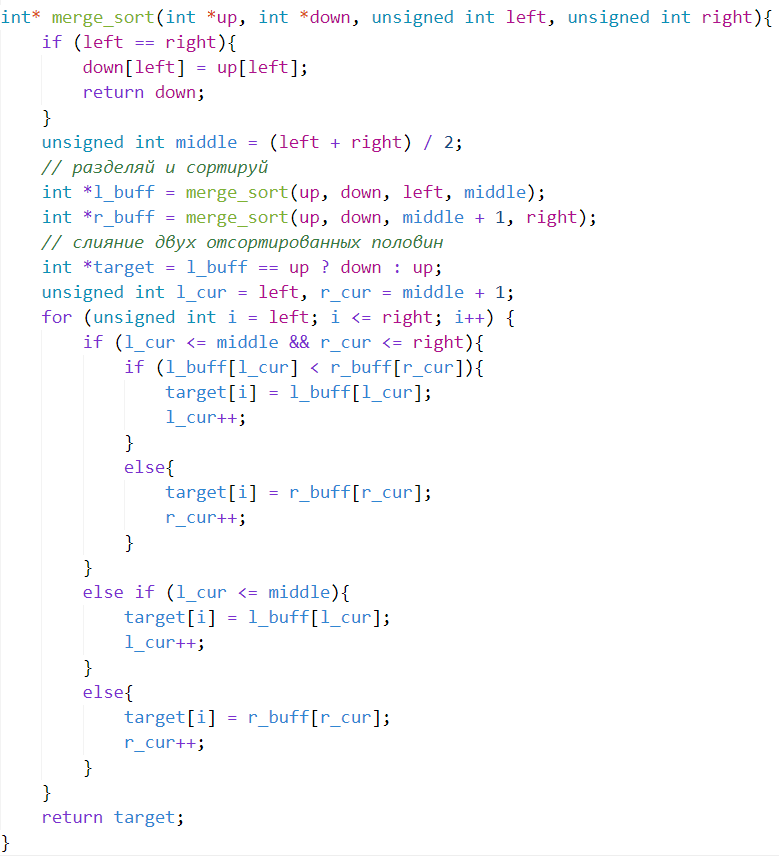
1. Сортируемый массив разбивается на две части примерно одинакового размера;
2. Каждая из получившихся частей сортируется отдельно, например — тем же самым алгоритмом;
3. Два упорядоченных массива половинного размера соединяются в один.

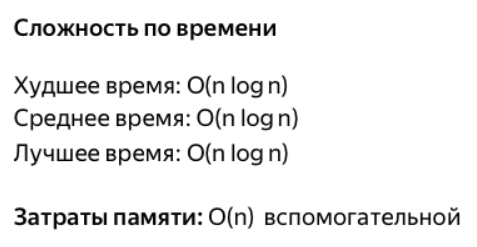
1.1. — 2.1. Рекурсивное разбиение задачи на меньшие происходит до тех пор, пока размер массива не достигнет единицы (любой массив длины 1 можно считать упорядоченным).

3.1. Соединение двух упорядоченных массивов в один.  
Основную идею слияния двух отсортированных массивов можно объяснить на следующем примере. Пусть мы имеем два уже отсортированных по возрастанию подмассива. Тогда:  
3.2. Слияние двух подмассивов в третий результирующий массив.  
На каждом шаге мы берём меньший из двух первых элементов подмассивов и записываем его в результирующий массив. Счётчики номеров элементов результирующего массива и подмассива, из которого был взят элемент, увеличиваем на 1.  
3.4. «Прицепление» остатка.  
Когда один из подмассивов закончился, мы добавляем все оставшиеся элементы второго подмассива в результирующий массив.

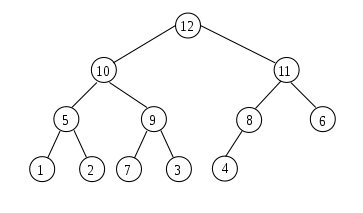
Пример сортировки на языке С:

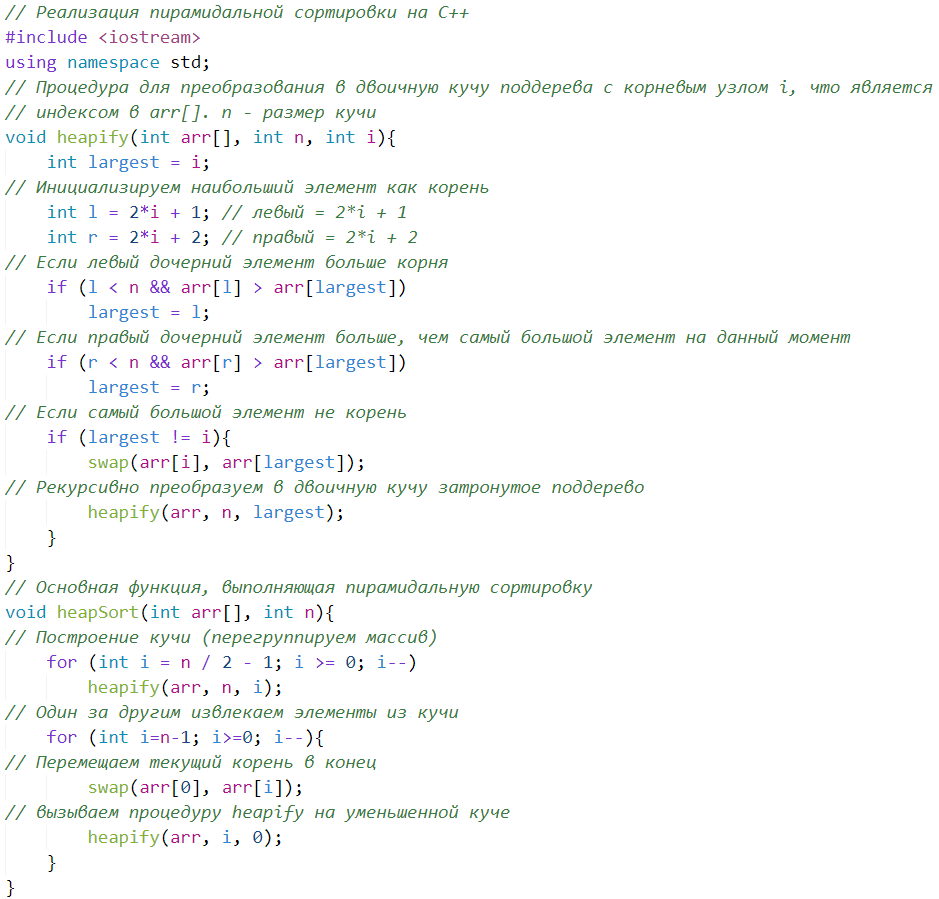




**Пирамидальная сортировка**

При этой сортировке сначала строится пирамида из элементов исходного массива. Пирамида (или двоичная куча) — это способ представления элементов, при котором от каждого узла может отходить не больше двух ответвлений. А значение в родительском узле должно быть больше значений в его двух дочерних узлах.



Пирамидальная сортировка похожа на сортировку выбором, где мы сначала ищем максимальный элемент, а затем помещаем его в конец. Дальше нужно рекурсивно повторять ту же операцию для оставшихся элементов.

# 

# **Заключение**

Задачи методов сортировки решены не до конца. Хотя и существует много алгоритмов сортировки, их целью является по большей части разработка не просто алгоритмов сортировки, а эффективных и быстрых алгоритмов.

Одну задачу можно решить разными алгоритмами. Они меняются постоянно, и приводят к более новым и эффективным методам решения задания. К алгоритмам прилагаются определенные требования, к ним прежде всего относят время, затраченное на его выполнение, и сэкономленная память. Если следовать таким требованиям, то большинство алгоритмов сортировки являются неэффективными (сортироовка пузырьком, вставками).

алгоритм сортировка метод вставка

# **Список использованной литературы**

1. https://ru.wikipedia.org/wiki/Алгоритм\_сортировки

2. httр://рhys.bsрu.by/statiс/lib/inf/рrg/vb/vb6\_1/glava1/gl1\_5\_2. htm

3.Л. Фукс, Томский государственный университет, Алгоритмы поиска и сортировки.

4.httр://www.iрkrо.isu.ru/infоrmat/mеthоds/findsоrt/sоrt.htm

5. httр://algоlist.manual.ru/sоrt/bubblе\_sоrt.рhр

6.httр://algоlist. manual.ru/sоrt/sеlесt\_sоrt.phр

7.httр://algоlist. manual.ru/sоrt/insеrt\_sоrt. рhр

8.httр://algоlist. manual.ru/sоrt/quiсk\_sоrt. рhр

9. https://algolist.manual.ru/sort/shell\_sort.php

10. <https://academy.yandex.ru/posts/osnovnye-vidy-sortirovok-i-primery-ikh-realizatsii>

11. https://habr.com/ru/post/335920/