Нетривиальная сортировка

ЛЕКЦИЯ 8

Сортировка Шелла

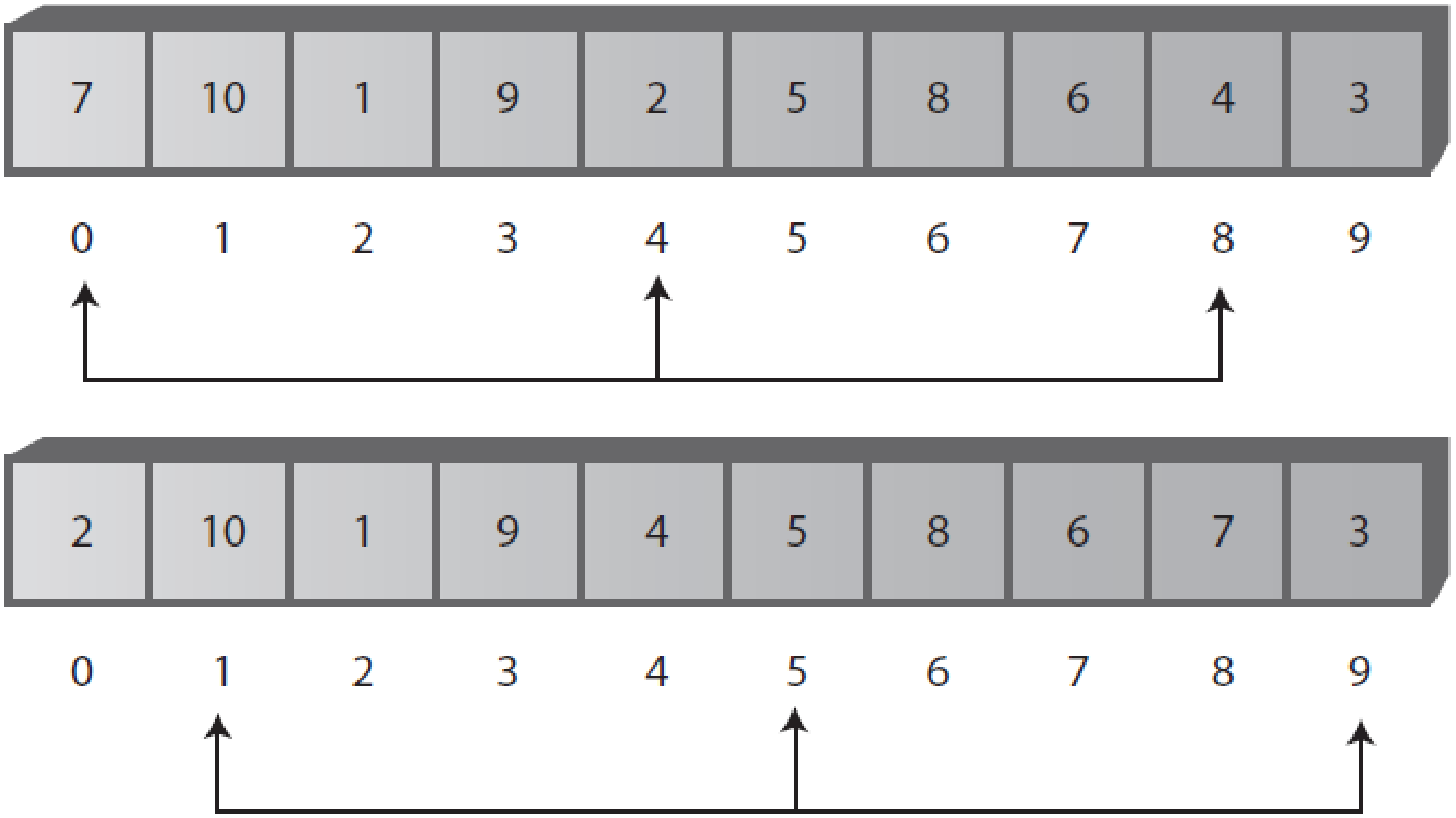
Алгоритм назван в честь Дональда Л. Шелла — специалиста в области компьютерных технологий, который опубликовал описание этого способа сортировки в 1959 году. Алгоритм Шелла основан на сортировке методом вставок, но он обладает новой особенностью, кардинально улучшающей скорость сортировки.

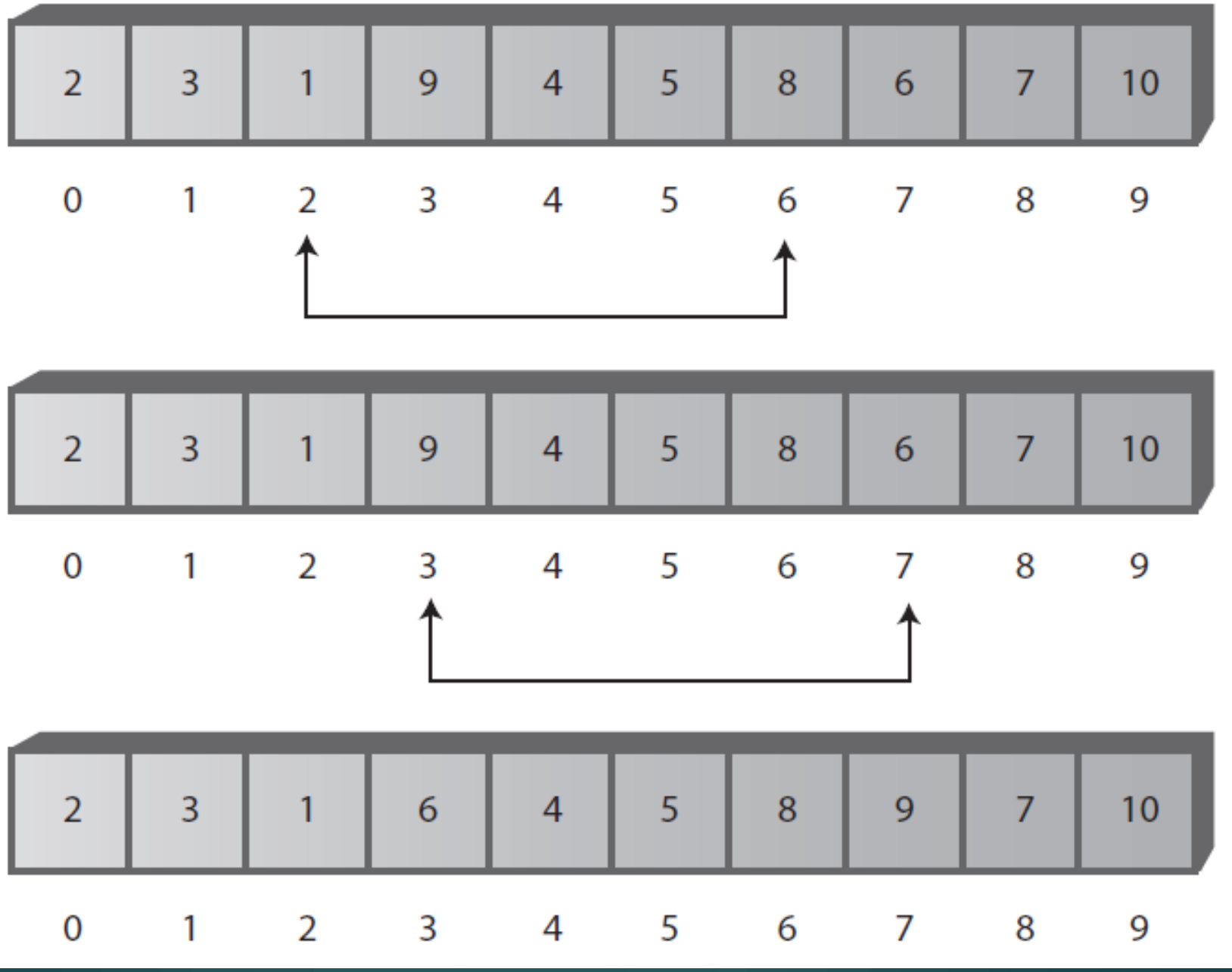
Сортировка Шелла хорошо подходит для массивов среднего размера — например, до нескольких тысяч элементов (в зависимости от конкретной реализации). По скорости она уступает быстрой сортировке и другим алгоритмам, выполняемым за время $O(N \times \log N)$, поэтому для очень больших объемов данных она не оптимальна.

N-сортировка

Сортировка Шелла выполняет «дальние» сдвиги, сортируя разобщенные элементы посредством сортировки методом вставок. После сортировки этих элементов сортируются элементы, находящиеся на меньшем расстоянии и т. д. Расстояние между элементами при такой сортировке называется *приращением* и традиционно обозначается буквой *h*.

Например, для 10-элементного массива с приращением 4. Здесь сортируются элементы 0, 4 и 8. После того как будут отсортированы элементы 0, 4 и 8, алгоритм сдвигается на одну ячейку и сортирует элементы 1, 5 и 9. Процесс продолжается до тех пор, пока все элементы не пройдут «4-сортировку», то есть все элементы, находящиеся на расстоянии 4-х ячеек, не будут отсортированы между собой.


7 10 1 9 2 5 8 6 4 3
0 1 2 3 4 5 6 7 8 9
2 10 1 9 4 5 8 6 7 3
0 1 2 3 4 5 6 7 8 9


2 3 1 9 4 5 8 6 7 10
0 1 2 3 4 5 6 7 8 9
2 3 1 9 4 5 8 6 7 10
0 1 2 3 4 5 6 7 8 9
2 3 1 6 4 5 8 9 7 10
0 1 2 3 4 5 6 7 8 9

N-сортировка

После полной 4-сортировки массив можно рассматривать как совокупность четырех подмассивов: (0,4,8), (1,5,9), (2,6) и (3,7), каждый из которых полностью отсортирован. Элементы подмассивов чередуются, но в остальном подмассивы полностью независимы друг от друга.

Обратите внимание: в этом конкретном примере в конце 4-сортировки ни один элемент не находится на расстоянии более двух ячеек от той позиции, где он находился бы при полной сортировке массива. Именно эта особенность — «почти» отсортированный массив — и является секретом сортировки Шелла. Создание чередующихся наборов элементов с внутренней сортировкой сводит к минимуму объем работы, которую необходимо проделать для завершения сортировки.

Интервальная последовательность

Ранее была продемонстрирована сортировка массива из 10 элементов с исходным интервалом в 4 ячейки. Для массивов большего размера предварительная сортировка должна начинаться с намного большего интервала, который затем последовательно сокращается вплоть до 1.

Например, в массиве из 1000 элементов может быть проведена 364-сортировка, затем 121-сортировка, 40-сортировка, 13-сортировка, 4-сортировка и, наконец, 1-сортировка. Серия чисел, используемых при генерировании интервалов (в данном примере 364, 121, 40, 13, 4, 1), называется ***интервальной последовательностью***.

Интервальная последовательность

Приведенная интервальная последовательность, приписываемая Кнуту , весьма популярна. В обратной форме, начиная с 1, она генерируется по рекурсивной формуле

$$h = 3 \times h + 1$$

- начальное значение *h* равно 1

```
public void shellSort()
{
    int inner, outer;
    long temp;
    int h = 1; // Вычисление исходного значения h

    while(h <= nElems/3)
        h = h*3 + 1; // (1, 4, 13, 40, 121, ...)

    while(h>0) // Последовательное уменьшение h до 1
    {
        // h-сортировка файла
        for(outer=h; outer<nElems; outer++){
            temp = theArray[outer];
            inner = outer;
                        // Первый подмассив (0, 4, 8)
            while(inner > h-1 && theArray[inner-h] >= temp){
                theArray[inner] = theArray[inner-h];
                inner -= h;
            }
            theArray[inner] = temp;
        }
        h = (h-1) / 3; // Уменьшение h
    }
}
```

Эффективность сортировки Шелла

До настоящего момента еще никому не удалось теоретически обосновать эффективность сортировки Шелла, если не считать некоторых частных случаев. Оценки, полученные на основании экспериментов, лежат в интервале от O (N^ 3/2) до O(N^ 7/6).

Разбиение

Разбиением данных называется такое разделение на две группы, при котором в одну группу входят все элементы со значением ключа выше заданного порога, а в другую — все элементы со значением ключа ниже заданного порога.

Эффективность алгоритма разбиения

Алгоритм разбиения выполняется за время $O(N)$

Быстрая сортировка

Бесспорно, быстрая сортировка является самым популярным алгоритмом сортировки, а это вполне понятно: в большинстве ситуаций он выполняется быстрее всех за время $O(N*\log N)$. (Это утверждение справедливо только для внутренней сортировки, выполняемой в памяти; при сортировке данных в файлах на диске другие алгоритмы могут оказаться более эффективными.) Быструю сортировку открыл Ч.Э.Р. Хоар в 1962 году.

Алгоритм быстрой сортировки

Алгоритм состоит из трех основных шагов:

1. Массив или подмассив разбивается на две группы: левую (с меньшими ключами) и правую (с большими ключами).

2. Рекурсивный вызов метода для сортировки левой группы.

3. Рекурсивный вызов метода для сортировки правой группы.

После разбиения все элементы левого подмассива меньше элементов правого подмассива. Если отсортировать левый подмассив и правый подмассив, то весь массив будет упорядочен.

Алгоритм быстрой сортировки

Аргументы метода recQuickSort() определяют левую и правую границы массива (или подмассива), который должен сортироваться при вызове. Метод сначала проверяет, не состоит ли массив только из одного элемента. Если условие выполняется, значит, массив уже отсортирован, и метод немедленно возвращает управление. Это базовое ограничение процесса рекурсии.

Алгоритм быстрой сортировки

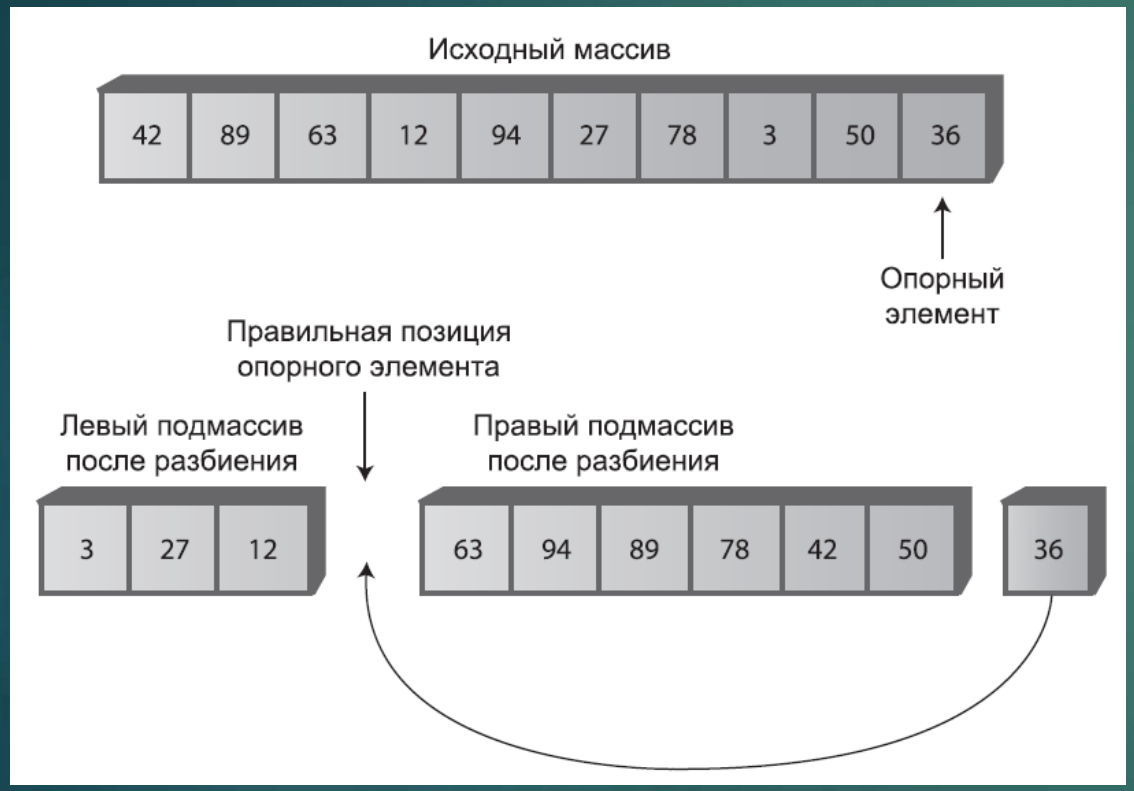
Если массив состоит из двух и более ячеек, то алгоритм вызывает метод partitionIt() для его разбиения. Метод возвращает *индекс разбиения* — индекс левого элемента правого подмассива (с бо´льшими значениями ключей). Это значение отмечает положение границы между подмассивами.

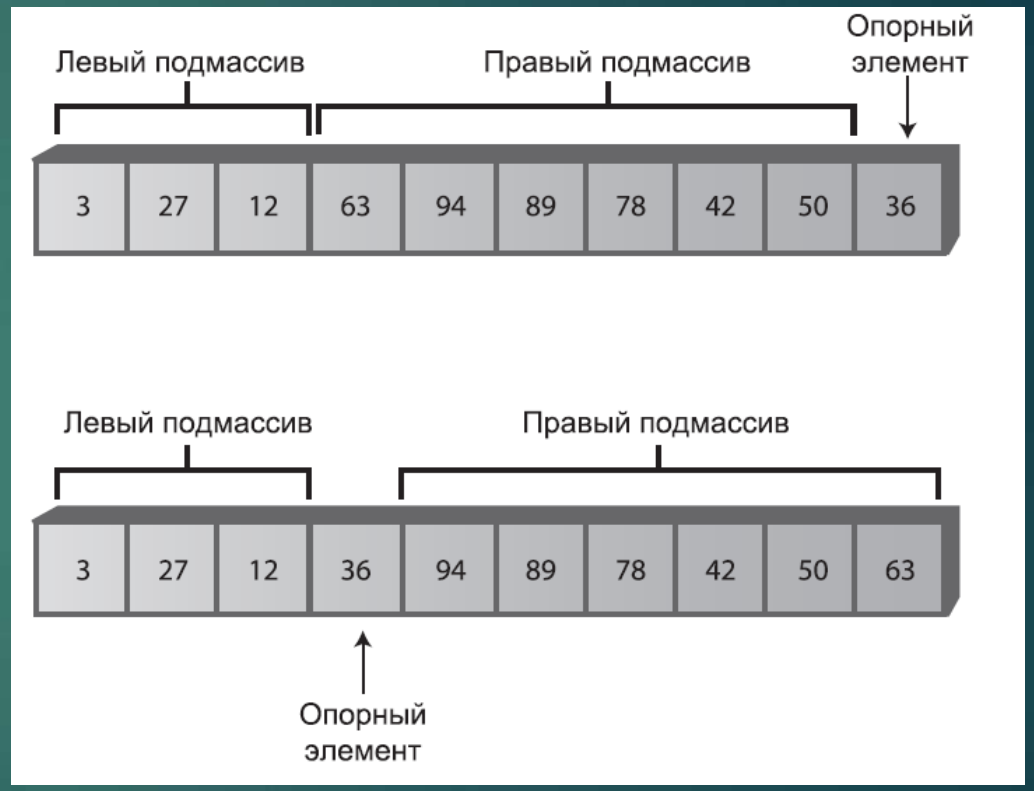
Выбор опорного значения

Какое опорное значение должен использовать метод partitionIt()? Несколько разумных идей:

- Опорное значение должно быть ключом существующего элемента; этот элемент называется опорным.
- Опорный элемент выбирается более или менее произвольно. Для простоты мы всегда будем выбирать элемент, находящийся у правого края массива.
- После разбиения опорный элемент, вставленный на границе между левым и правым подмассивом, всегда находится в своей окончательной позиции сортировки.

Опорный элемент и подмассивы





Поразрядная сортировка

Алгоритм поразрядной сортировки будет рассматриваться в контексте обычной десятичной системы счисления, так как она более наглядна. Однако эффективная реализация алгоритма использует двоичную систему счисления, чтобы пользоваться преимуществами быстрых компьютерных операций с битами. Мы рассмотрим поразрядную сортировку вместо похожей, но несколько более сложной *обменной поразрядной сортировки*. Сортировка основана на раздельном анализе каждой цифры ключа, начиная с последней (наименее значащей) цифры.

Алгоритм поразрядной сортировки

1. Все элементы данных делятся на 10 групп в соответствии со значением последней цифры.
2. Элементы всех 10 групп собираются в единую последовательность: сначала идут все элементы с ключами, завершающимися цифрой 0, затем все элементы с ключами, завершающимися цифрой 1 и так далее до 9. Мы будем называть этот шаг «вспомогательной сортировкой».
3. При второй вспомогательной сортировке все данные снова делятся на 10 групп, но на этот раз в соответствии со значением предпоследней цифры (цифры десятков). Деление не должно нарушать порядка предыдущей сортировки. Иначе говоря, в каждой из 10 групп порядок элементов остается таким же, как после шага 2 (вспомогательные сортировки должны быть устойчивыми).

Алгоритм поразрядной сортировки

4. Все 10 групп снова объединяются. Сначала идут элементы, у которых предпоследняя цифра равна 0, затем элементы с предпоследней цифрой 1 и так далее до 9.
5. Процесс повторяется для всех остальных цифр. Если некоторые ключи содержат меньше цифр, чем другие, считается, что их старшие разряды заполнены 0.

Пример поразрядной сортировки

421 240 035 532 305 430 124 // Исходный массив

(240 430) (421) (532) (124) (035 305) // Сортировка по цифре единиц

(305) (421 124) (430 532 035) (240) // Сортировка по цифре десятков

(035) (124) (240) (305) (421 430) (532) // Сортировка по цифре сотен

035 124 240 305 421 430 532 // Отсортированный массив

Итоги

- Сортировка Шелла применяет сортировку методом вставки к элементам, разделенных большими интервалами; затем интервалы сокращаются и т. д.
- Выражением «n-сортировка» обозначается сортировка каждого *n*-го элемента.
- Для определения интервалов сортировки в алгоритме Шелла используется последовательность чисел, называемая *интервальной последовательностью*.
- Часто применяемая интервальная последовательность генерируется по рекурсивной формуле $h = 3 \times h + 1$, где исходное значение h равно 1.
- *Разбиением* массива называется его деление на два подмассива, один из которых содержит элементы с ключом ниже заданной величины, а другой — элементы с ключом большим либо равным этой величины.
- *Опорным значением* называется значение, определяющее принадлежность элемента к той или иной группе в процессе разбиения. Элементы, меньшие опорного значения, попадают в левую группу; бо´льшие элементы попадают в правую группу.