

Лекция №3. Простая сортировка.

Пузырьковая сортировка



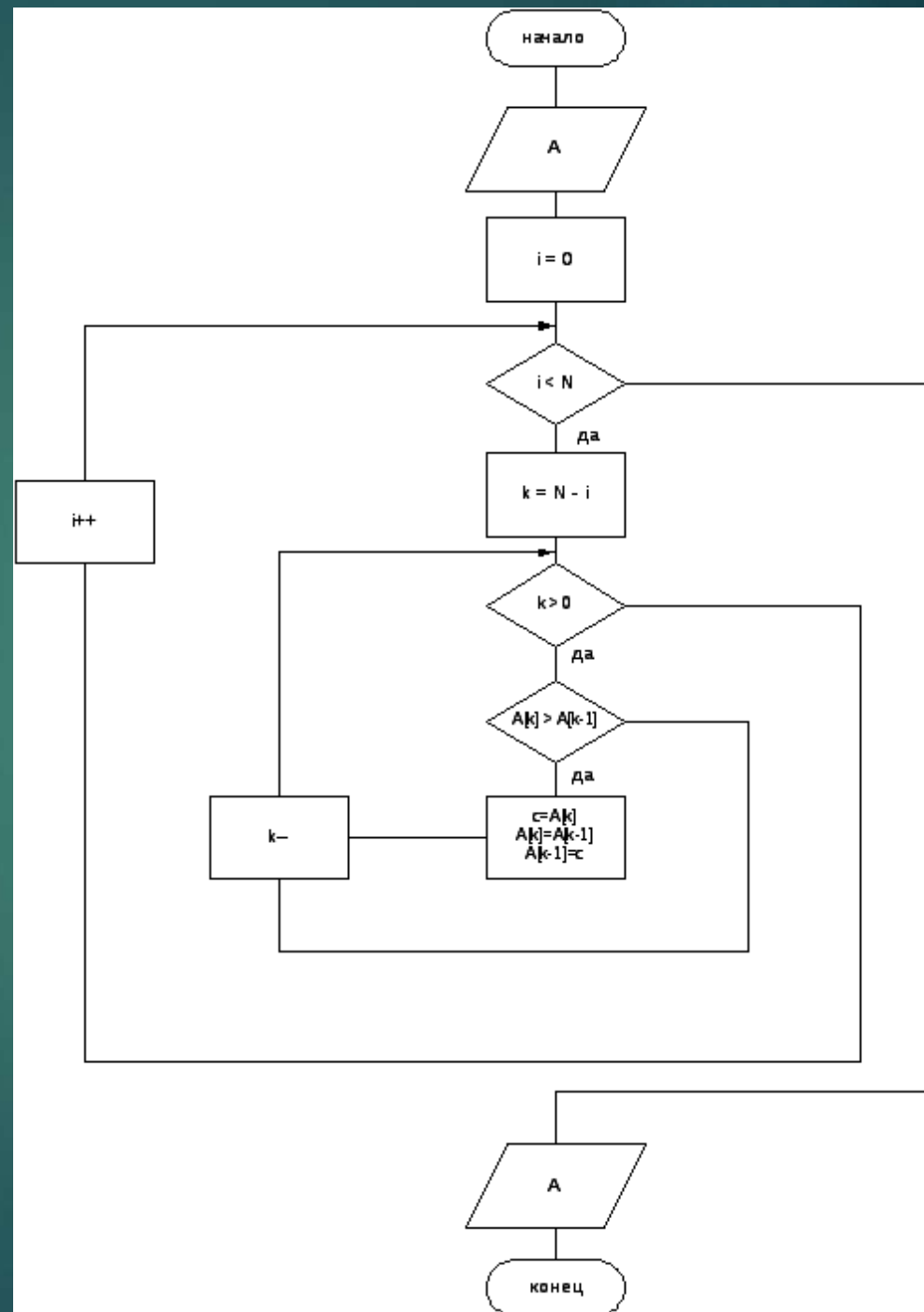
Алгоритм пузырьковой сортировки : вы подходите к левому краю шеренги (позиция 0) и сравниваете два элемента в позициях 0 и 1. Если левый элемент (позиция 0) больше, вы меняете их местами. Если больше правый элемент, они остаются на своих местах. Затем вы переходите на одну позицию вправо и сравниваете элементы в позициях 1 и 2. И снова, если левый элемент больше, вы меняете их местами.

Пузырьковая сортировка

Сортировка выполняется по следующим правилам:

1. Сравнить два элемента.
2. Если левый больше, поменять их местами.
3. Перейти на одну позицию вправо.
4. После того как первый отсортированный элемент окажется на своем месте, вернуться к началу массива.

Сортировка пузырьком



Задача №1

Реализация пузырьковой сортировки на
языке Java

Решение

```
public void bubbleSort() {  
    int out, in;  
    for(out=nElems-1; out>1; out--) // Внешний цикл (обратный)  
        for(in=0; in<out; in++) // Внутренний цикл (прямой)  
            if( a[in] > a[in+1] ) // Порядок нарушен?  
                swap(in, in+1); // Поменять местами  
}
```

```
private void swap(int one, int two){  
    long temp = a[one];  
    a[one] = a[two];  
    a[two] = temp;  
}
```

Инварианты

Во многих алгоритмах задействованы условия, которые остаются неизменными в ходе выполнения алгоритма. Такие условия называются инвариантами. Умение распознавать инварианты поможет понять суть алгоритма. Инварианты также могут пригодиться в ходе отладки; вы можете периодически проверять истинность инвариантов и выдавать сообщение об ошибке в случае их нарушения

Сложность пузырьковой сортировки

В общем виде, если массив состоит из N элементов, на первом проходе выполняются $N-1$ сравнений, на втором — $N-2$ и т. д. Сумма такого ряда вычисляется по формуле:


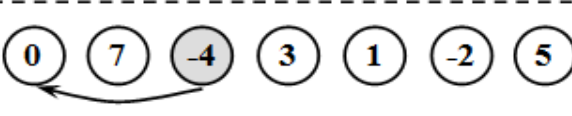
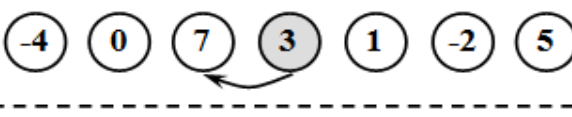

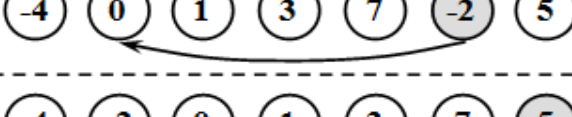
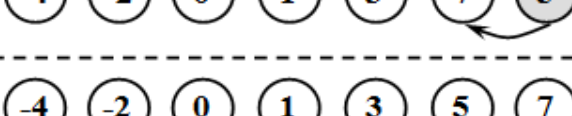


$$(N-1) + (N-2) + (N-3) + \dots + 1 = N \times (N-1)/2.$$

Если $N = 10$, то сумма равна 45 ($10 \times 9/2$).

Таким образом, алгоритм выполняет около $N^2 / 2$ сравнений (на -1 можно не обращать внимания, особенно при больших N).

Количество как перестановок, так и сравнений пропорционально N^2 . Так как константы в O -синтаксисе не учитываются, можно сказать, что пузырьковая сортировка выполняется за время $O(N^2)$.

Сортировка методом выбора

	
1 проход	
2 проход	
3 проход	
4 проход	
5 проход	
6 проход	
РЕЗУЛЬТАТ	

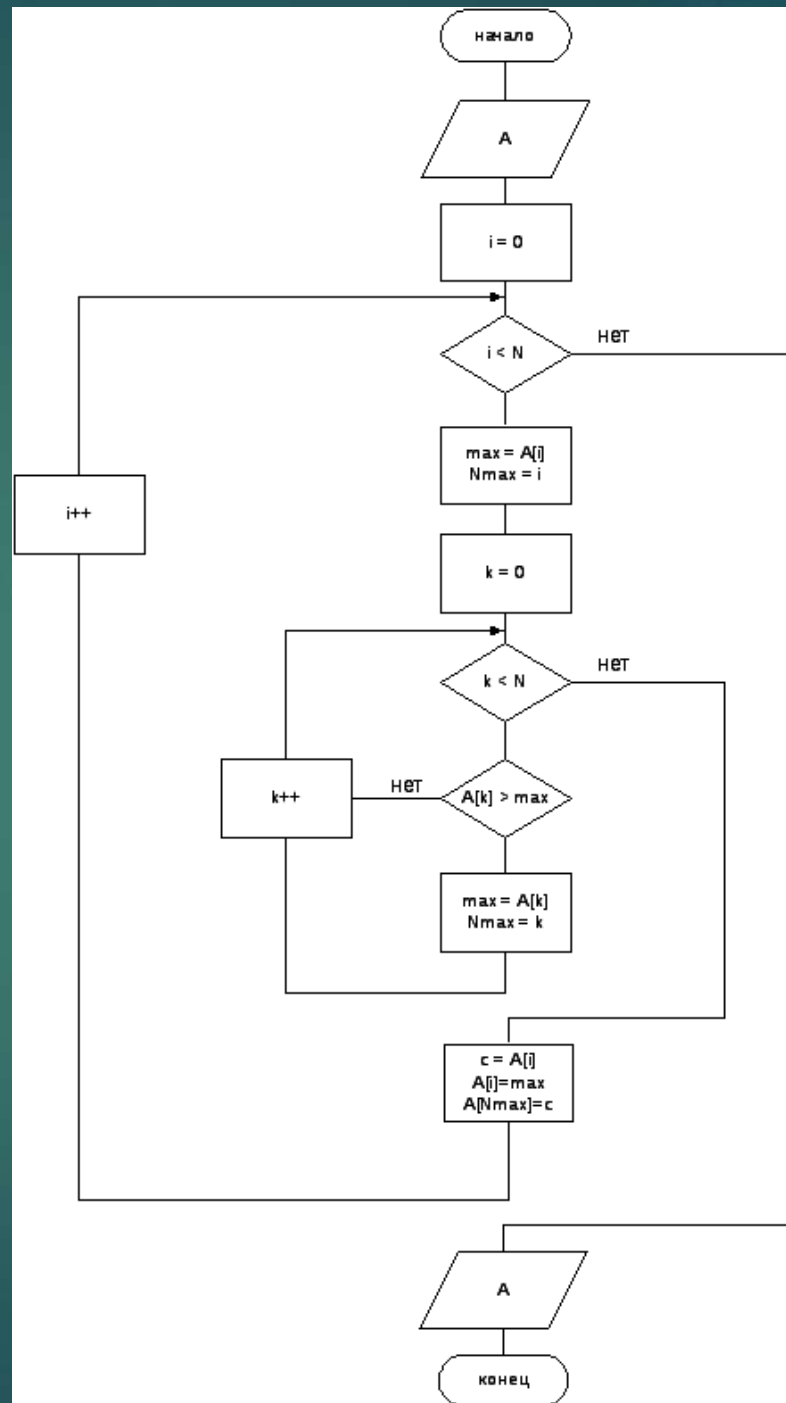
В сортировке методом выбора вы последовательно перебираете все элементы и выбираете (отсюда и название) самый маленький из них. Этот элемент меняется местами с тем, который стоит в крайней левой позиции (0). Левый элемент отсортирован, и в дальнейшем уже перемещаться не будет. Обратите внимание: в этом алгоритме отсортированные элементы собираются слева (нижние индексы), тогда как при пузырьковой сортировке они собираются справа.

Сортировка выбором

Сортировка выполняется по следующим правилам:

1. Выбрать наименьший (наибольший) элемент в массиве.
2. Поменять его местами с элементом стоящим на первом месте.
3. Перейти к пункту 1, выбирая наименьший (наибольший) элемент, считая, что массив уменьшился «слева» на 1.

Сортировка выбором



Сложность сортировки методом выбора

Сортировка методом выбора выполняет такое же количество сравнений, что и пузырьковая сортировка:

$$N \times (N-1)/2.$$

Сортировка методом выбора

Алгоритм сортировки методом выбора превосходит пузырьковую сортировку по характеристикам — количество необходимых перестановок сокращается с $O(N^2)$ до $O(N)$. К сожалению, количество сравнений остается равным $O(N^2)$.

Сортировка методом вставки

В большинстве случаев сортировка методом вставки является лучшим из элементарных алгоритмов сортировки, описанных в этой главе. Она также выполняется за время $O(N^2)$, но работает примерно вдвое быстрее пузырьковой сортировки, а в обычных ситуациях немного быстрее сортировки методом выбора. Кроме того, сортировка методом вставки не слишком сложна, хотя ее алгоритм немного сложнее двух других. Она часто используется на завершающей стадии более сложных алгоритмов, например быстрой сортировки.

Сортировка вставкой

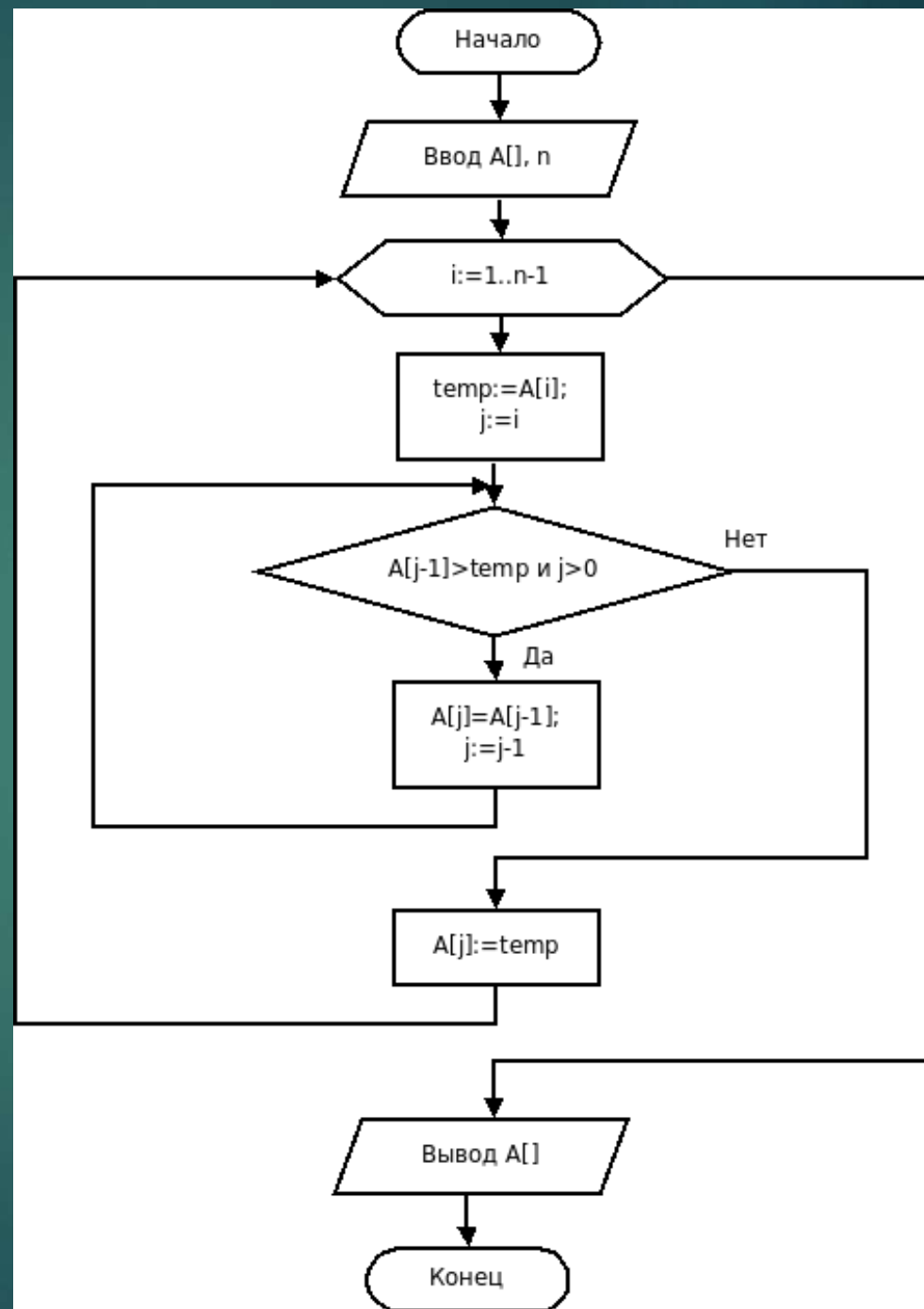
Сортировка выполняется по следующим правилам:

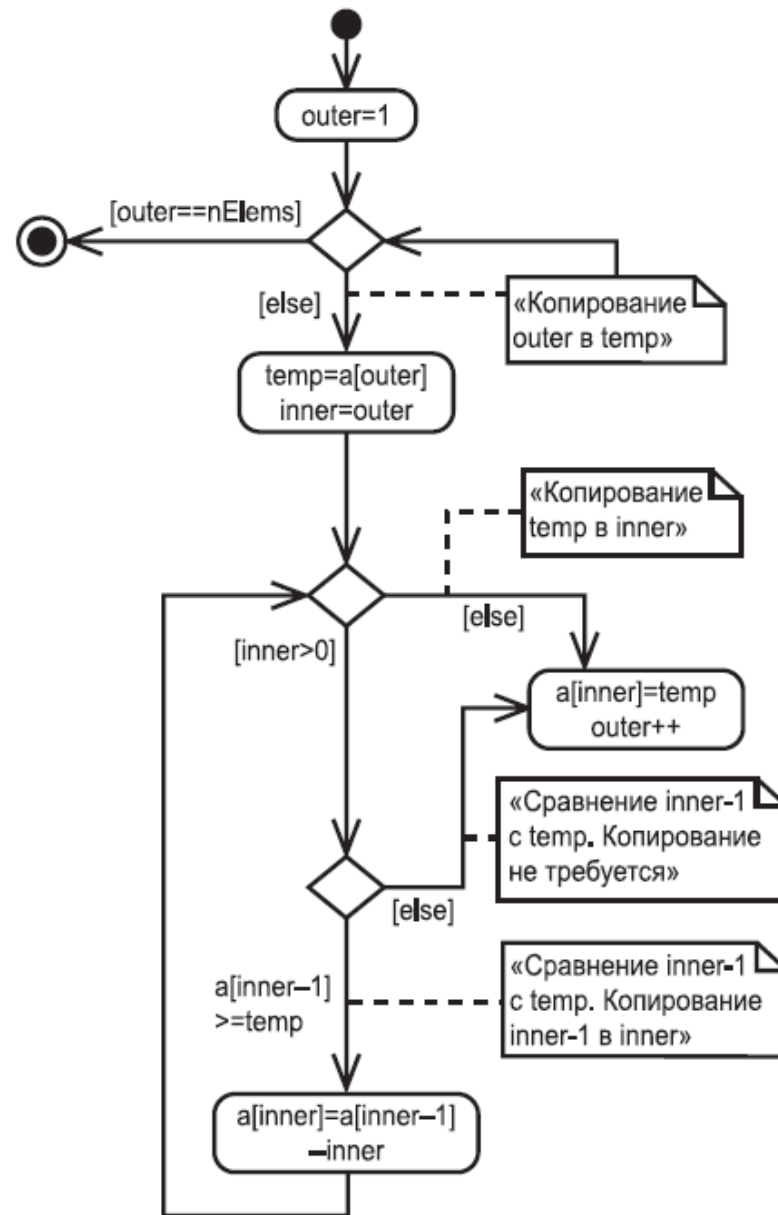
1. Сравнивать элементы массива, пока элемент $N+1$ не нарушит правило сортировки
2. «Вынуть» данный элемент из массива.
3. Сдвигать последующие элементы к началу массива, сравнивать их с «вынутым»
4. Вставить элемент на подходящее место.
4. Вернуться к той ячейке из которой был «вынут» элемент, продолжить проверку.

Сортировка методом вставки

В середине элементов устанавливается воображаемый маркер. Элементы слева от маркера частично отсортированы; это означает, что каждый из них больше своего соседа слева. Однако это вовсе не означает, что элементы занимают свои итоговые позиции, потому что они, возможно, будут перемещены в результате вставки других, еще не отсортированных элементов. Обратите внимание: ситуация частичной сортировки не возникает в ходе пузырьковой сортировки и сортировки методом выбора. В этих алгоритмах подгруппа элементов данных полностью упорядочена в любой конкретный момент времени; в сортировке методом вставки подгруппа элементов упорядочена только частично.

Сортировка вставкой





Лексикографические сравнения

Метод `compareTo()` возвращает разные целочисленные значения в зависимости от относительного лексикографического (то есть алфавитного) расположения объекта `String`, для которого вызван метод, и объекта `String`, переданного в аргументе.

<code>s2.compareTo(s1)</code>	Возвращаемое значение
<code>s1 < s2</code>	<0
<code>s1 = s2</code>	0
<code>s1 > s2</code>	>0

Устойчивость сортировки

В некоторых случаях важен относительный порядок элементов данных, имеющих одинаковые ключи. Например, данные работников могут быть упорядочены в алфавитном порядке фамилий. (То есть поле фамилии используется в качестве ключа сортировки.) Затем данные потребовалось отсортировать по почтовым индексам, но так, чтобы все элементы с одинаковым индексом остались отсортированными по фамилиям. Алгоритм должен сформировать новые группы, однако в полученных группах должен сохраниться исходный порядок следования элементов. Некоторые алгоритмы сортировки сохраняют этот вторичный порядок; они называются устойчивыми (или стабильными).

Использование сортировок

Не стоит использовать данные методы сортировок для массивов размером больше 1000

Большинство современных ПК способны разобраться с подобными методами сортировки достаточно быстро и допускается однократное использование данных методов

Сортировку пузырьком используют «по памяти»

Сортировка выбором «оптимальна» для однократной сортировки неупорядоченного массива

Сортировка вставкой используется в реальных проектах, но только на завершающих этапах при работе с частично отсортированными данными

Лабораторная работа №2

Реализация сортировок методом вставки и методом выбора, исследование эффективности алгоритмов.

Системный таймер в Java:

```
Long start = System.currentTimeMillis();
```

```
<...>
```

```
Long end = System.currentTimeMillis();
```

Спасибо!