НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ЯДЕРНЫЙ УНИВЕРСИТЕТ «МИФИ»
Кафедра информатики и процессов управления (№17)

Дисциплина «Информатика (основы программирования)»

Методические указания

Тематическое занятие 9

Сортировка массива.

Содержание

Постановка задачи 1
Сортировка и поиск *1*
Методы сортировки *2*
Метод вставки (включения) 2
Принцип метода *2*
Описание алгоритма *2*
Способы улучшения *3*
Метод выбора (выделения) 4
Принцип метода и алгоритм *4*
Способы улучшения *4*
Метод обмена (“пузырька”) 5
Принцип метода и алгоритм *5*
Способы улучшения *5*

Постановка задачи

Сортировка и поиск

Сортировка – процесс упорядочивания однородного набора данных *по возрастанию* (или *убыванию*) значения какого-либо признака. Если набор данных содержит элементы с одинаковым значением признака, то говорят о сортировке *по неубыванию* (или *невозрастанию*).

Сортировка – одно из наиболее важных действий над массивами в системах сбора и поиска информации. В отсортированном массиве можно гораздо быстрее найти нужную информацию, чем в неотсортированном.

Далее признаком, по которому проводится сортировка, будем считать само значение элемента массива.

При решении задачи сортировки обычно выдвигается требование **минимального использования дополнительной памяти**, т.е. использование дополнительных массивов *недопустимо.* Поэтому при сортировке одномерного массива его элементы **меняются местами** таким образом, что их значения оказываются упорядоченными.

Существует множество различных алгоритмов сортировки, которые значительно отличаются друг от друга по ***быстродействию*** (скорости работы). Для оценки быстродействия алгоритмов, как правило используют два показателя – количество операций ***присваивания*** и ***сравнения***.

Методы сортировки

Все методы сортировки можно разделить на две большие группы: ***прямые*** и ***улучшенные*** методы сортировки.

Прямые методы в свою очередь делятся на три подгруппы по принципу, лежащему в основе метода:

- метод ***вставки*** *(****включения****) –****Insertion*** *sort;*
- метод ***выбора*** *(****выделения****) –* ***Selection*** *sort;*
- метод ***обмена*** *(****“пузырьковая”*** сортировка) – ***Bubble*** *sort.*

Улучшенные методы основываются на тех же принципах, что и прямые, но используют некоторые оригинальные идеи для ускорения процесса сортировки.

Прямые методы на практике используются довольно редко, т.к. имеют относительно **низкое** быстродействие. Однако они хорошо показывают суть основанных на них улучшенных методов. Кроме того, в некоторых случаях (небольшой размер массива и/или особое исходное расположение элементов) прямые методы могут даже превосходить улучшенные.

Метод вставки (включения)

Принцип метода

Массив разделяется на две части: отсортированную и неотсортированную. Элементы из неотсортированной части поочередно выбираются и вставляются в отсортированную часть так, чтобы не нарушить в ней упорядоченность элементов.

Определяем, где текущий элемент должен находиться в упорядоченной части, и вставляем его туда.

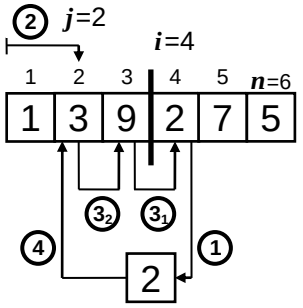
В начале работы алгоритма в качестве отсортированной части массива принимается только один первый элемент, а в качестве неотсортированной – все остальные.

Описание алгоритма

Для массива из n элементов алгоритм метода вставки состоит из $(n-1)$-го прохода, каждый из которых содержит четыре последовательных действия:

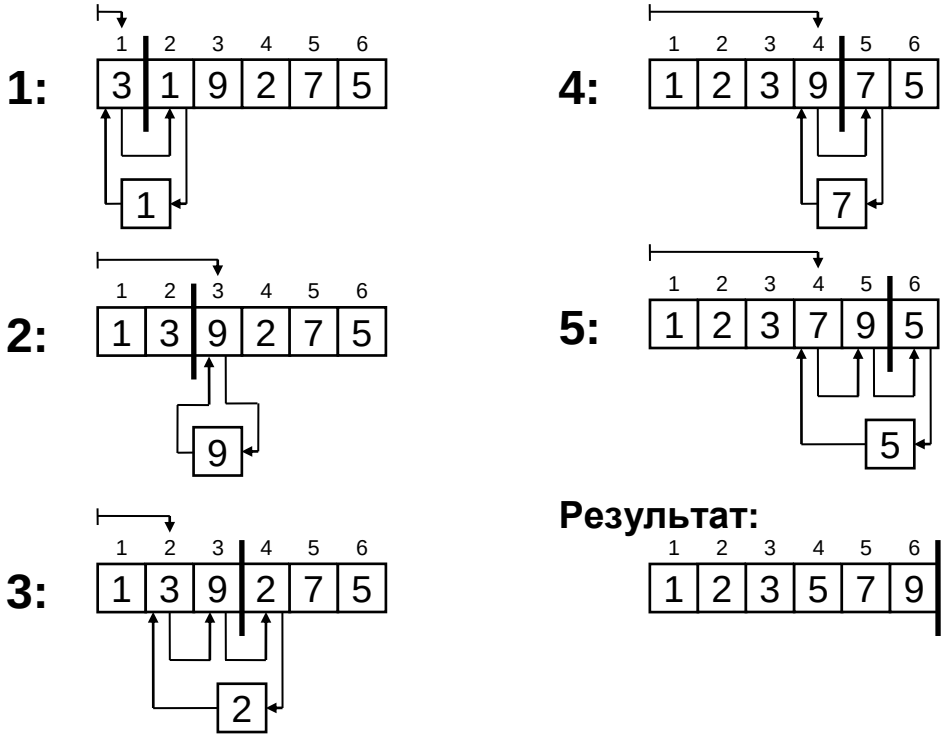
1. взятие очередного i-го элемента массива из неотсортированной части и сохранение его в дополнительной переменной;
2. поиск позиции j в отсортированной части массива, для которой вставка сохраненного элемента не нарушит упорядоченности;
3. сдвиг элементов массива от $(i-1)$-го до j-го вправо, чтобы освободить найденную позицию вставки;
4. вставка сохраненного элемента в найденную j-ю позицию.

Описанные действия одного прохода можно представить на следующей схеме:



① – сохранение
② – поиск места вставки
③ – сдвиг
④ – вставка

На следующей схеме представлен пример сортировки массива из 6-и элементов за 5 проходов алгоритма вставки:



Для реализации данного алгоритма можно предложить несколько способов поиска позиции вставки. Например, для сортировки массива по возрастанию, – это может быть перебор всех элементов отсортированной части (начиная с первого) до нахождения элемента, значение которого больше сохраненного в дополнительной переменной (это и будет ***j***-й элемент).

Способы улучшения

Вкратце алгоритм метода вставки можно сформулировать так: для $i=2..n$ каждый элемент массива $A[i]$ помещается на свое место в упорядоченной ранее части массива $A[1], \ldots, A[i-1]$, при этом, если необходимо, происходит сдвиг элементов массива.

Модифицировать алгоритм метода вставки для повышения его эффективности можно, изменив способ поиска позиции вставки следующим образом.

На момент вставки элемента $A[i]$ элементы массива $A[1], \ldots, A[i-1]$ уже отсортированы. Нужно выбрать средний элемент и сравнить его с $A[i]$. Если $A[i]$ меньше этого элемента, то поиск места вставки следует продолжать в левой половине упорядоченной части массива, иначе – в правой.

Метод выбора (выделения)

Принцип метода и алгоритм

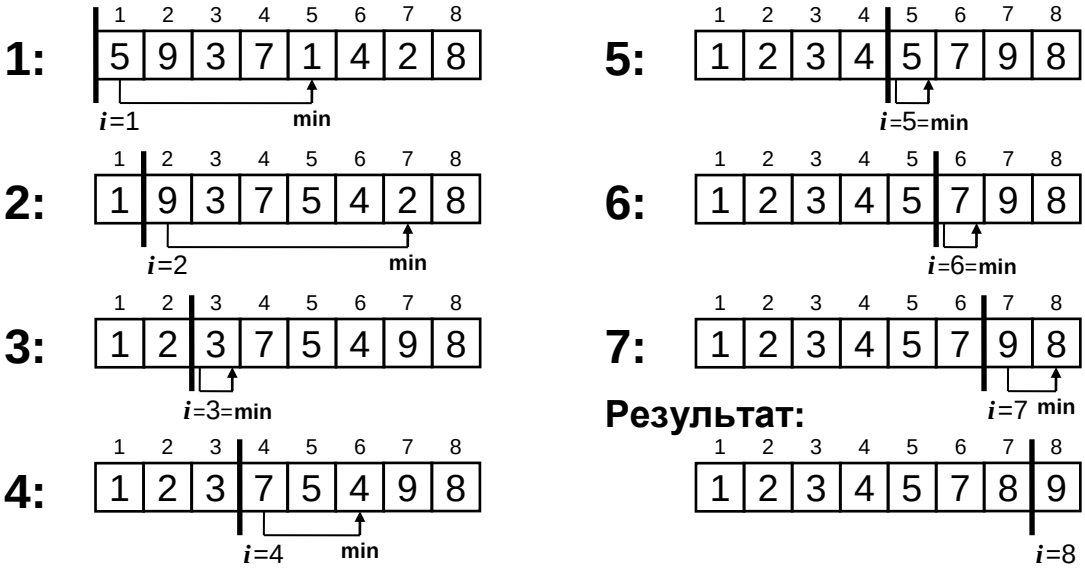
В массиве, содержащем ***n*** элементов, находим (выбираем) элемент с минимальным значением на интервале от ***1***-го до ***n***-го (последнего) элемента и меняем его местами с первым элементом.

На втором шаге находим элемент с минимальным значением на интервале от ***2***-го до ***n***-го элемента и меняем его местами со вторым элементом.

И так далее для всех элементов до ***(n–1)***-го. Всего потребуется ***(n–1)*** проход.

Ищем наименьший (или наибольший) элемент и помещаем его в начало (или конец) упорядоченной части.

Следующая схема демонстрирует пример сортировки массива из 8-и элементов за 7 проходов алгоритма выбора:



Способы улучшения

Вкратце алгоритм метода выбора можно сформулировать так: необходимо менять местами очередной элемент с максимальным среди неупорядоченной части массива, то есть $A[i] = max\{ A[j],\ j = i..n \},\ i = 1..(n-1)$.

Модифицировать алгоритм метода выбора, чтобы повысить его эффективность для массивов с большим количеством многократно повторяющихся элементов, можно следующим образом.

Если в неупорядоченной части массива найти первый максимальный элемент $A[k]$, то среди следующих за ним элементов $A[k]$, …, $A[n]$ могут содержаться много равных $A[k]$. Их следует сразу помещать в отсортированную часть массива, следом за $A[k]$.

Метод обмена (“пузырька”)

Принцип метода и алгоритм

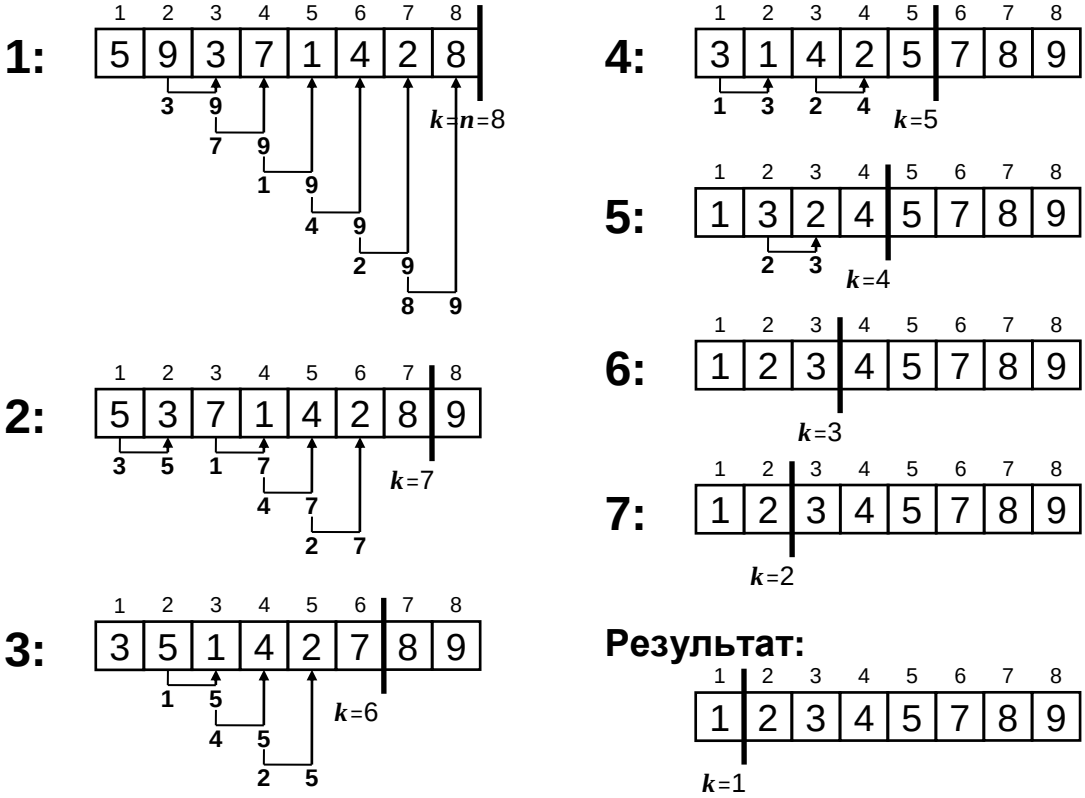
Слева направо поочередно сравниваются два соседних элемента, и если их взаиморасположение не соответствует заданному условию упорядоченности, то они меняются местами. Далее берутся два следующих соседних элемента и так далее до конца массива.

После одного такого прохода на последней ***n***-й позиции массива будет стоять максимальный элемент (“всплыл” первый “пузырек”). Поскольку максимальный элемент уже стоит на своей последней позиции, то второй проход обменов выполняется до ***(n–1)***-го элемента. И так далее.

Всего потребуется ***(n–1)*** проход.

Для каждой пары индексов производим обмен, если элементы расположены не по порядку.

Схема работы метода “пузырька” показана на следующем примере сортировки массива из 8-и элементов за 7 проходов:



Способы улучшения

Вкратце алгоритм метода “пузырьковой” сортировки можно сформулировать так: необходимо многократно переставлять два соседних элемента, нарушающих порядок, пока весь массив не будет упорядочен.

Модифицировать алгоритм метода “пузырька”, чтобы повысить его эффективность, можно следующим образом.

Во-первых, нужно исключить лишние просмотры элементов массива, если они есть. Например, для сортировки по возрастанию массива **12,3,5,7,9,10** достаточно одного просмотра.

Во-вторых, требуется устранить несимметричность метода по отношению к различным исходным данным, меняя поочередно направления просмотров, слева направо и справа налево. Например, массив **5,7,9,10,12,3** должен быть упорядочен по возрастанию за два просмотра.