Практическое занятие № 2 «Программирование алгоритмов сортировки одномерных массивов на языке С++»

Учебные цели:

- получение умений и навыков создания и отладки программ для обработки одномерных массивов;
- получение умений и навыков программирования алгоритмов сортировки массивов данных;
- получение умений и навыков создания и отладки программ с циклической структурой.

Воспитательные цели:

- формировать диалектико-материалистическое мировоззрение;
- формировать навыки самостоятельности и дисциплинированности;
- стимулировать активную познавательную деятельность обучаемых, способствовать формированию у них творческого мышления.

Категория слушателей: 2,3 курс.

Время: 90 мин.

Место проведения: компьютерный класс.

Материально-техническое обеспечение:

1) персональный компьютер *IBM PC* с операционной системой Windows XP; 2) среда разработки приложений *Visual C++.NET.*

ПЛАН ПРАКТИЧЕСКОГО ЗАНЯТИЯ

Учебные вопросы	Время, мин
Вступительная часть	5
1. Сортировка одномерных массивов в С++	15
2. Выполнение индивидуального задания	65
Заключительная часть	5

Элементы теории

Алгоритмы сортировки информации образуют основу для огромного большинства прикладных программ. Сортировка информации − это одна из стандартных функций, возникающих в процессе решения задач.

Сортировка данных − это процесс изменения порядка расположения элементов в некоторых упорядоченных структурах данных таким образом, чтобы обеспечить возрастание или убывание числового значения элемента

данных или определенного числового параметра, связанного с каждым элементом данных (ключа), при переходе от предыдущего элемента к последующему. То есть для любой пары чисел определены отношения «больше» или «меньше».

Для переменных символьного типа понятия «возрастание» и «убывание» относятся к значениям машинного кода, используемого для представления символов в памяти компьютера. Так как все буквенные символы располагаются в таблице кодов по алфавиту, то сортировка слов текста всегда приводит к их упорядочению в алфавитной последовательности.

Сортировка данных используется для эффективного решения других задач при программировании. Для упорядоченной совокупности данных быстро и легко решается задача, как поиск и отбор информации по заданному условию.

Существует много алгоритмов, обеспечивающих решение задачи сортировки. Одни из них обладают низким быстродействием, другие обладают очень высокой эффективностью и практически используются в современных компьютерных системах.

Различают два вида сортировки данных:

- сортировка данных, расположенных в оперативной памяти компьютера *(внутренняя сортировка*);
- сортировка данных, расположенных на внешних запоминающих устройствах (*внешняя сортировка*).

Сформулируем постановку задачи сортировки данных для внутренней сортировки одномерного числового массива по возрастанию.

Имеется одномерный массив чисел, состоящий из n элементов: A[n]. Переставить элементы массива так, чтобы их значения располагались в порядке возрастания. Другими словами, для любой пары элементов A[i] и A[i+1] выполняется неравенство вида: A[i] <= A[i+1].

В этой задаче однозначно определяется структура данных для внутренней сортировки (одномерный массив) и порядок упорядочения элементов. Ключом для определения порядка элементов является само числовое значение элемента массива. Результатом решения задачи должна быть программа сортировки массива одним или несколькими методами.

При разработке программы можно воспользоваться различными алгоритмами. Наиболее известными являются следующие:

- метод сортировки обменами («пузырьковая» сортировка);
- метод сортировки вставками;
- метод сортировки выбором элемента;
- метод разделения (алгоритм «быстрой» сортировки метод Хоора);
- метод «пирамиды» (метод Уильямса-Флойда);
- метод «счетчика».

Главным показателем качества алгоритма внутренней сортировки является скорость сортировки. Алгоритмы «пирамиды», «счетчика» и «быстрой» сортировки обеспечивают высокую скорость сортировки и

находят широкое практическое применение. Недостатком «быстрой» сортировки является возможность резкого увеличения трудоемкости при «неблагоприятном» исходном порядке элементов в массиве. С другой стороны, метод «пирамиды» в целом отстает по скорости от «быстрой» сортировки. Метод «счетчика» является быстрым, однако не рационально использует оперативную память компьютера, поскольку требует введения дополнительного массива.

Выбор в пользу того или иного алгоритма может быть сделан при условии тщательного статистического анализа реальной задачи и это является достаточно важной проблемой в программировании.

Алгоритмы сортировки

Алгоритм сортировки обменами («пузырьковая» сортировка). Метод «пузырька» один из самых простых методов внутренней сортировки. Суть алгоритма состоит в последовательном просмотре массива от конца к началу или от начала к концу и сравнении каждой пары элементов между собой. При этом «неправильное» расположение элементов устраняется путем их перестановки. Процесс просмотра и сравнения элементов повторяется до просмотра всего массива. При сортировке по возрастанию «легкие» элементы с меньшим значением как бы «всплывают» к началу массива подобно тому, как это делают пузырьки воздуха в стакане с водой − отсюда и происходит популярное название алгоритма.

«Пузырьковая» сортировка имеет очень плохие временные характеристики. Она имеет только учебно-исторический интерес и не может быть рекомендована для практического использования.

В таблице 1 подробно расписан процесс упорядочивания элементов в массиве. Нетрудно заметить, что для преобразования массива, состоящего из n элементов, необходимо просмотреть его n-1 раз, каждый раз уменьшая диапазон просмотра на один элемент. Схема описанного алгоритма приведена на рис. 1. Обратите внимание на то, что для перестановки элементов (блок 4) используется буферная переменная *B*, в которой временно хранится значение элемента, подлежащего замене.

Таблица 1

Процесс упорядочивания элементов в массиве по возрастанию

Номер элемента	**1**	**2**	**3**	**4**	**5**
Исходный массив	7	3	5	4	2
Первый просмотр	3	5	4	2	7
Второй просмотр	3	4	2	5	7
Третий просмотр	3	2	4	5	7
Четвертый просмотр	2	3	4	5	7

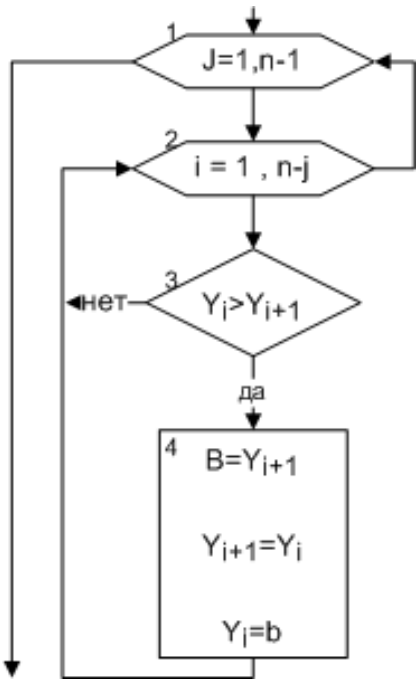

Рис. 1. Сортировка массива пузырьковым методом

Алгоритм сортировки вставками. *Принцип метода*. Массив разделяется на две части: отсортированную и несортированную. Элементы из несортированной части поочередно выбираются и вставляются в отсортированную часть так, чтобы не нарушить в ней упорядоченность элементов.

Алгоритм.

1. В качестве отсортированной части массива принимается первый элемент.

2. Берется очередной элемент из несортированной части. Его значение сохраняется в дополнительной переменной.

3. Осуществляется поиск позиции в отсортированной части массива, в которой присутствие взятого элемента не нарушит упорядоченности элементов

4. Для освобождения найденной позиции элементы сдвигаются вправо, после чего на это место вставляется сохраненное ранее значение.

Пункты 2–4 выполняются до полной упорядоченности массива.

Схема алгоритма сортировки вставками представлена на рис. 2. Организуем цикл для просмотра всех элементов массива, начиная со второго (блок 4). Сохраним значение текущего **i**-го элемента во вспомогательной переменной **X**, так как оно может быть потеряно при сдвиге элементов (блок 5) и присвоим переменной **j** значение индекса предыдущего **(i-1)**-го элемента массива (блок 6). Далее движемся по массиву влево в поисках элемента меньшего, чем текущий и пока он не найден сдвигаем элементы вправо на одну позицию. Для этого организуем цикл (блок 7), который прекратиться, как только будет найден элемент меньше текущего. Если такого элемента в массиве не найдется и переменная **j** станет равной нулю, то это будет

означать, что достигнута левая граница массива, и текущий элемент необходимо установить в первую позицию. Смещение элементов массива вправо на одну позицию выполняется в блоке 8, а изменение счетчика **j** в блоке 9. Блок 10 выполняет вставку текущего элемента в соответствующую позицию.

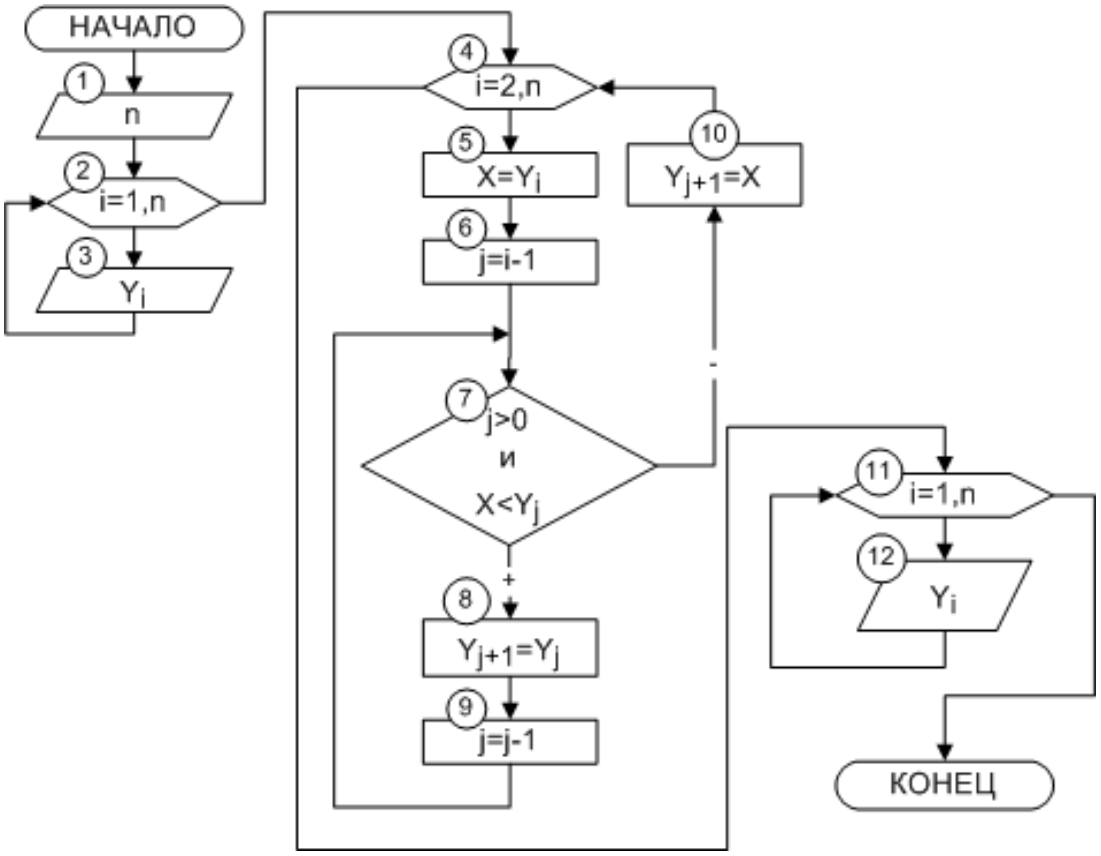

Рис. 2. Сортировка массива вставкой

Преимущества сортировки методом вставки:

- эффективен на небольших наборах данных, на наборах данных до десятков элементов может оказаться лучшим;
- эффективен на наборах данных, которые уже частично отсортированы;
- это устойчивый алгоритм сортировки (не меняет порядок элементов, которые уже отсортированы);
- может сортировать список по мере его получения.

Минусом же является высокая сложность алгоритма: $O(n^2)$.

Алгоритм сортировки выбором элемента представлен в виде схемы на рис. 3. Найдем в массиве самый большой элемент (блоки 3-7) и поменяем его местами с последним элементом (блок 8). Повторим алгоритм поиска максимального элемента, уменьшив количество просматриваемых элементов на единицу (блок 9), и поменяем его местами с предпоследним элементом

(блоки 3-7). Описанную выше операцию поиска проводим до полного упорядочивания элементов в массиве. Так как в блоке 9 происходит изменение переменной n, то в начале алгоритма ее значение необходимо сохранить (блок 1).

Примечание. Для упорядочивания массива по убыванию необходимо перемещать минимальный элемент.

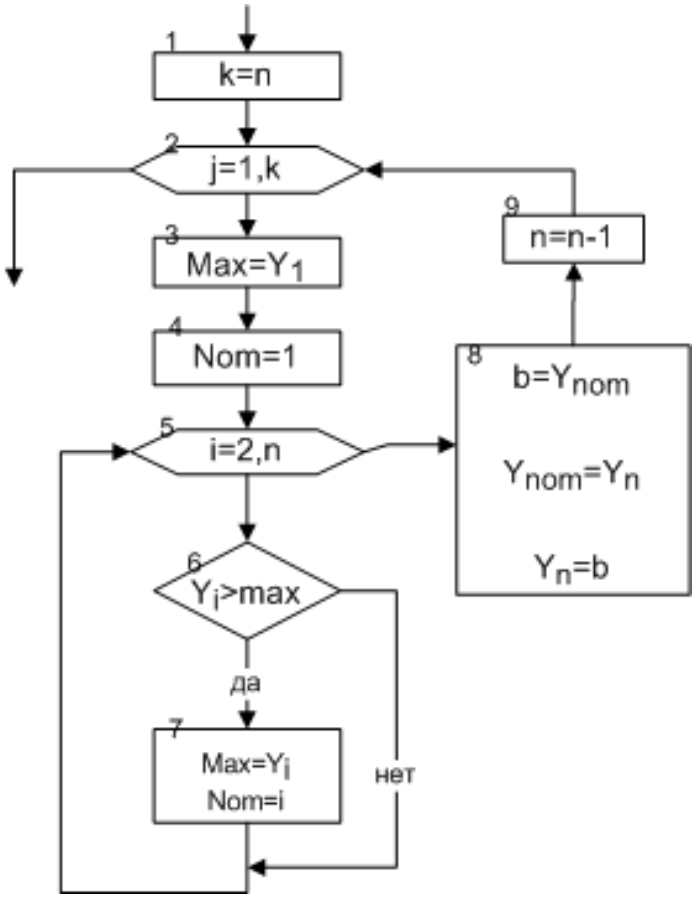

Рис. 3. Сортировка массива выбором наибольшего элемента

По сравнению с алгоритмами вставки и «пузырька» алгоритм сортировки выбором элемента в большинстве случаев может оказаться более быстрым.

Вопросы для самопроверки

1. Что понимается под сортировкой массива?
2. Чем отличается сортировка по убыванию от сортировки по невозрастанию?
3. Сформулировать идею сортировки массива методом простого выбора.
4. Почему время выполнения одного шага метода простого выбора прямо пропорционально размеру неупорядоченной части массива?
5. Сформулировать идею сортировки массива методом обмена.
6. Как поменять местами два элемента массива?

7. Что такое вложенные циклы?
8. Сформулировать идею сортировки массива методом вставки.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОТРАБОТКЕ УЧЕБНЫХ ВОПРОСОВ

Практическое задание

Задание 1. В соответствии с Вашим вариантом составьте и отладьте программу обработки одномерного массива.

1. Отсортировать элементы одномерного массива по возрастанию, используя метод сортировки обменами, начиная с первого элемента массива, большего числа введенного с клавиатуры.
2. Отсортировать элементы одномерного массива по убыванию, используя метод сортировки вставками, начиная с первого элемента массива, большего числа введенного с клавиатуры.
3. Отсортировать элементы одномерного массива по возрастанию, используя метод сортировки выбором элемента, начиная с первого элемента массива, большего числа введенного с клавиатуры.
4. Отсортировать элементы одномерного массива по убыванию, используя метод сортировки обменами, начиная с первого элемента массива, меньшего числа введенного с клавиатуры.
5. Отсортировать элементы одномерного массива по возрастанию, используя метод сортировки вставками, начиная с первого элемента массива, меньшего числа введенного с клавиатуры.
6. Отсортировать элементы одномерного массива по убыванию, используя метод сортировки выбором элемента, начиная с первого элемента массива, меньшего числа введенного с клавиатуры.
7. Отсортировать элементы одномерного массива по возрастанию, используя метод сортировки обменами, начиная с 3-его и заканчивая 10-ым элементом массива.
8. Отсортировать элементы одномерного массива по убыванию, используя метод сортировки вставками, лежащие между 4-ым и 11-ым элементом массива.
9. Отсортировать элементы одномерного массива по возрастанию, используя метод сортировки выбором элемента, лежащие между 2-ым и 7-ым элементом массива.
10. Отсортировать элементы одномерного массива по убыванию, используя метод сортировки вставками, лежащие между 5-ым и 15-ым элементом массива.

Контрольные вопросы

1. Какое количество проходов потребуется при сортировке массива методом выбора?
2. Какое количество проходов потребуется при сортировке массива методом вставки?
3. Какое количество проходов потребуется при сортировке массива методом обмена?
4. В чем сходство и отличие методов простого выбора и простого обмена?
5. Перечислите преимущества и недостатки сортировки методом вставки.
6. Опишите алгоритм сложной сортировки вставками (сортировка Шелла).
7. Сформулируйте идею алгоритма быстрой сортировки (метод Хоора).