ІНСТРУКТИВНА КАРТКА
ПРАКТИЧНОГО ЗАНЯТТЯ №8

Тема: Основні алгоритми сортування

Мета: Метою роботи є вивчення алгоритмів сортування.

Обладнання: ноутбуки , програмне забезпечення: Dev-Cpp C++

ЛІТЕРАТУРА

1. ПРОГРАМУВАННЯ МОВОЮ C++ Ю. А. Бєлов Т. О. Карнаух Ю. В. Коваль, ВПЦ «Київський університет», 2012
2. http://tmb.org.ua/new/index.php?option=com_content&view=article&id=135:2011-11-30-09-10-05&catid=36:4--&Itemid=81
3. http://lib.selyam.net/docs/6000/index-2710.html
4. http://idndist.lp.edu.ua/moodle/library/books/0007/index8_8.html

Завдання

Напишить програму, в якій:

1. Згенерувати одновимірний (або двовимірний, в залежності від варіанту) масив цілих чисел розмірністю згідно варіанту.
2. Елементи масиву задати випадковим чином в діапазоні 0 ... 50.
3. Виконайте друк цього масиву на екран.
4. Виконайте обробку масиву відповідно до варіанту. При написанні програми використати вказаний алгоритм сортування.
5. Виконайте друк перетвореного масиву на екран.

Завдання

1. Відсортувати одновимірний масив довжиною N = 50 за спаданням методом простого обміну.
2. Відсортувати одновимірний масив довжиною N = 55 за зростанням методом вибору.
3. Є одномірний масив довжиною N = 33. Упорядкувати масив методом вибору таким чином, щоб елементи, що знаходяться на парних позиціях розташовувалися за спаданням, а на непарних позиціях - по зростанню.
4. Відсортувати одновимірний масив довжиною N = 67 методом простого обміну за зростанням. Після упорядкування в кожній групі повторюваних елементів залишити тільки один, а решта видалити. Наприклад, якщо відсортований масив має вигляд: 1 2 2 4 4 4 8 9, кінцевий масив буде мати вигляд: 1 2 4 8 9
5. Є двовимірний масив розмірністю NxN, де N = 10. Відсортувати всі стовпці методом простого обміну так, щоб елементи в них розташовувалися за зростанням.

6. Є одномірний масив довжиною N = 45. Відсортувати за спаданням методом вибору ті елементи масиву, які є парними числами.
7. Є одномірний масив довжиною N = 38. Упорядкувати масив методом простого обміну таким чином, щоб елементи, що знаходяться на парних позиціях розташовувалися по зростанню, а на непарних позиціях - за спаданням.
8. Є одномірний масив довжиною N = 70. Відсортувати за зростанням за допомогою методу вибору ті елементи масиву, які розташовуються на непарних позиціях.
9. Є двовимірний масив розмірністю NxN, де N = 9. Відсортувати всі рядки методом простого обміну так, щоб елементи в них розташовувалися за спаданням.
10. Є двовимірний масив розмірністю NxN, де N = 8. Відсортувати всі рядки методом вибору так, щоб елементи в них розташовувалися за зростанням.
11. Є двовимірний масив розмірністю NxN, де N = 10. Відсортувати всі рядки методом вибору так, щоб елементи в них розташовувалися за спаданням.
12. Відсортувати одновимірний масив довжиною N = 29 за спаданням методом вибору.
13. Відсортувати одновимірний масив довжиною N = 31 по зростанню методом простого обміну.
14. Є двовимірний масив розмірністю NxN, де N = 9. Відсортувати всі стовпці методом вибору так, щоб елементи в них розташовувалися за зростанням.
15. Відсортувати за спаданням одновимірний масив довжиною N = 44 методом вибору. Після упорядкування в кожній групі повторюваних елементів залишити тільки один, а решта видалити. (Дивись приклад з вар. 4).
16. Є двовимірний масив розмірністю NxN, де N = 10. Відсортувати всі стовпці методом простого обміну так, щоб елементи в них розташовувалися за спаданням.
17. Є одномірний масив довжиною N = 32. Відсортувати за зростанням за допомогою методу простого обміну ті елементи масиву, які розташовуються на непарних позиціях.
18. Є одномірний масив довжиною N = 26. Відсортувати за спаданням за допомогою методу вибору ті елементи масиву, які є непарними числами.
19. Є двовимірний масив розмірністю NxN, де N = 8. Відсортувати всі рядки методом простого обміну так, щоб елементи в них розташовувалися за зростанням.

20. Є одномірний масив довжиною N = 25. Упорядкувати масив методом вибору таким чином, щоб елементи, що знаходяться на парних позиціях розташовувалися по зростанню, а на непарних позиціях - за спаданням.
21. Є одномірний масив довжиною N = 27. Упорядкувати масив методом простого обміну таким чином, щоб елементи, що знаходяться на парних позиціях розташовувалися за спаданням, а на непарних позиціях - по зростанню.

Контрольні питання

1. Для чого потрібні алгоритми сортування?
2. З яких основних частин складається будь-який алгоритм сортування?
3. Які основні параметри алгоритмів сортування ви знаєте?
4. Поясніть принцип роботи сортування вибором.
5. Поясніть принцип роботи сортування методом простого обміну.

Найпростіші алгоритми сортування

Алгоритми сортування

Алгоритмом сортування називається алгоритм для впорядкування деяких елементів множини. Зазвичай алгоритмом сортування вважають алгоритм упорядкування елементів за зростанням або спаданням.

Практично кожен алгоритм сортування можна розбити на 3 етапи:

 порівняння, що визначає впорядкованість пари елементів;

 перестановку, міняє місцями пару елементів;

 власне алгоритм що сортує, який здійснює порівняння і перестановку елементів до тих пір, поки всі елементи множини не будуть впорядковані.

Алгоритми сортування мають велике практичне застосування. Їх можна зустріти там, де мова йде про обробку та зберігання великих обсягів інформації. Деякі завдання обробки даних вирішуються простіше, якщо дані заздалегідь упорядкувати.

Існує величезна кількість різних алгоритмів сортування. Універсального, найкращого алгоритму сортування на даний момент не існує. Однак, маючи приблизні характеристики вхідних даних, можна підібрати метод, який працює оптимальним чином при даних умовах. Для цього необхідно знати параметри, за якими буде проводитися оцінка алгоритмів.

Основні параметри алгоритмів сортування

Час сортування - основний параметр, що характеризує швидкодію алгоритму.

Необхідна пам'ять - один з параметрів, який характеризується тим, що ряд алгоритмів сортування вимагають виділення додаткової пам'яті для тимчасового зберігання даних. При оцінці пам'яті що використовується не буде враховуватися місце, яке займає вихідний масив даних та витрати які не залежать від вхідної послідовності, наприклад, на зберігання коду програми.

Стійкість - це параметр, який відповідає за те, що сортування не змінює взаємного розташування рівних елементів.

Природність поведінки - параметр, якій вказує на ефективність методу при обробці вже відсортованих, або частково відсортованих даних. Алгоритм поводиться природно, якщо враховує цю характеристику вхідної послідовності і працює краще.

Сортування вибором

Алгоритм сортування за зростанням:

1. Знайти мінімальне значення в поточному списку.

2. Знайдене мінімальне значення міняється місцем з елементом на першій позиції.
3. Повторюємо сортування, виключивши з розгляду вже відсортований перший елемент, тобто, починаючи з другої позиції.

Послідовність кроків при *n=7* показана на малюнку нижче.

Алгоритм не використовує додаткової пам'яті: всі операції відбуваються "на місці".

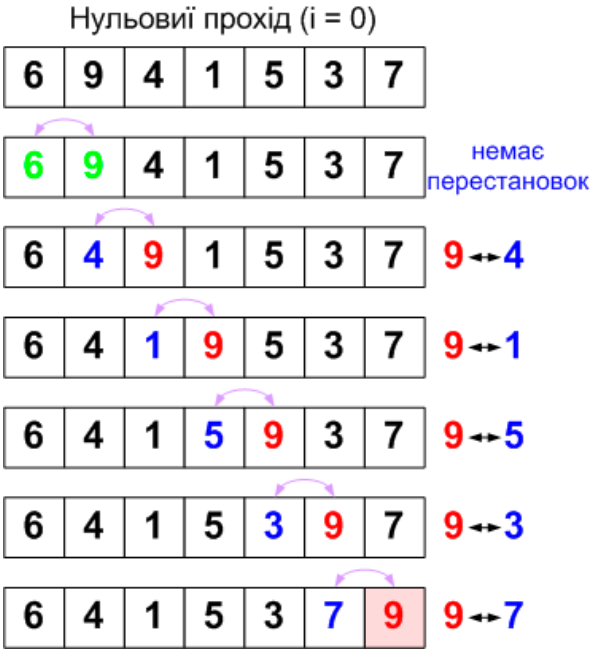
Метод простого обміну (метод бульбашки).

Алгоритм полягає в повторюваних проходах по сортованому масиву. За кожен прохід елементи послідовно порівнюються попарно і, якщо порядок у парі невірний, виконується обмін елементів. Проходи по масиву повторюються до тих пір, поки на черговому проході не виявиться, що обміни більше не потрібні, що означає - масив відсортований. При проході алгоритму, елемент, що стоїть не на своєму місці, «спливає» до потрібної позиції як бульбашка у воді, звідси і назва алгоритму.

Алгоритм сортування методом бульбашки:

1. Порівнюємо перший і другий елементи масиву. Якщо перший елемент більший, ніж другий, то міняємо їх місцями.
2. Порівнюємо другий і третій елементи масиву. Якщо другий елемент більший, ніж третій, то міняємо їх місцями.
3. ...

4. Порівнюємо передостанній *(N-1)* і останній *(N)* елементи масиву. Якщо передостанній елемент більший, ніж останній, то міняємо їх місцями.



В результаті останнім елементом в масиві у нас виявиться найбільший елемент.

Другий крок сортування методом бульбашки - повторюємо вищевказані дії для частини масиву, починаючи з першої позиції до *N-1*:

1. Порівнюємо перший і другий елементи масиву. Якщо перший елемент більший, ніж другий, то міняємо їх місцями.
2. Порівнюємо другий і третій елементи масиву. Якщо другий елемент більший, ніж третій, то міняємо їх місцями.
3. ...
4. Порівнюємо елемент (*N-2)* і елемент (*N-1*) масиву. Якщо (*N-2*)-й елемент більший, ніж елемент (*N-1*), то міняємо їх місцями.

В результаті передостанній елемент в масиві у нас теж буде на своєму, "відсортованому" місці.

На наступних кроках сортування методом бульбашки вищевказані дії повторюються для частини масиву, починаючи з 1 позиції до (*N-2*) (крок 3), а потім для діапазону 1 .. (*N-3*) і так далі до діапазону 1 .. 2.

Після завершення останнього кроку масив буде відсортований за зростанням.

Середнє число порівнянь та обмінів мають квадратичний порядок зростання, звідси можна зробити висновок, що алгоритм бульбашки досить повільний, проте він простий і його можна покращити простими засобами.

По-перше, розглянемо ситуацію, коли на якому-небудь з проходів не відбулося жодного обміну. Це означає, що всі пари розташовані в правильному порядку, так що масив вже відсортований. Одна з можливостей поліпшення алгоритму полягає в запам'ятовуванні, чи проводився на даному проході обмін. Якщо ні - алгоритм закінчує роботу.

Процес поліпшення можна продовжити, якщо запам'ятовувати не тільки сам факт обміну, але і індекс останнього обміну *k*. Дійсно: всі пари сусідніх елементів з індексами, більшими за *k*, вже розташовані в потрібному порядку. Подальші проходи можна закінчувати на індексі *k*, замість того щоб рухатися до встановленої заздалегідь верхньої межі.