Лабораторна робота №3

АЛГОРИТМИ СОРТУВАННЯ ДАНИХ

Мета роботи: навчитися складати програми для сортування масивів даних з використанням алгоритмів gnome sort, insert sort.

Теоретичні відомості

Сортуванням називається процес обробки інформації, результатом якого є впорядкування певних даних за одним, або кількома критеріями. За рахунок сортування збільшується швидкість пошуку та доступу до даних, що зберігаються у великих масивах інформації. Процес сортування відіграє важливу роль у системах обробки інформації. Сортування може здійснюватися над іменами, числами, записами, тощо.

Сортування використовується для вирішення різних задач. Одним із прикладів застосування операції сортування в телекомунікаціях є відновлення початкової послідовності переданих мережею зв'язку пакетів інформації. В процесі передавання інформації за допомогою протоколу зв'язку IP (IP - Internet Protocol) необхідно розбивати великі блоки інформації на малі. Ця проблема виникає внаслідок того, що у протоколів на канальному рівні максимальний розмір блоку інформації, який можна передати одним пакетом (MTU — maximum transmission unit) є меншим за розмір блоку який надходить від протоколу вищого рівня. Наприклад, для технології Ethernet, MTU складає 1500 байт інформації, в той час як розмір блоку корисного навантаження протоколу IP є значно більшим. Таким чином, необхідно вирішувати проблему узгодження процесу передавання інформації з протоколами канального рівня. Для вирішення цієї проблеми здійснюється операція фрагментації, яка передбачає розбиття одного блоку інформації до розмірів MTU протоколу нижчого рівня. На приймальній стороні здійснюється зворотна операція — дефрагментація. Вона забезпечує відновлення початкової структури блоку інформації. Оскільки, пакети поширюються мережею зв'язку різними шляхами, послідовність їхнього надходження до вузла є різною. Саме це зумовлює необхідність сортування пакетів для правильного відтворення переданої інформації в процесі дефрагментації.

Іншим прикладом необхідності застосування операції сортування є обробка даних, які зберігаються в таблицях баз даних. Прикладом таких даних

в телекомунікаціях можуть бути IP-адреси вузлів мережі, кількість переданого трафіку, номери портів, ідентифікатори користувачів, тощо.

Сортування інформації застосовується для розв'язку різних задач та може здійснюватися з використанням різних алгоритмів. Оскільки, задача сортування масивів даних є актуальною, вона досліджувалася різними вченими. В результаті було запропоновано велику кількість різних алгоритмів, які мають свої переваги та недоліки.

Вимоги до алгоритмів сортування

Для забезпечення сортування великих об'ємів даних до алгоритмів ставляться різні вимоги, які є пов'язаними між собою. До таких вимог слід віднести:

• Надійність роботи алгоритму полягає у здатності алгоритму виконувати правильне сортування даних (згідно заданого критерію). Для підвищення швидкодії процесу сортування необхідно уникати зайвих етапів перестановки елементів із однаковими значеннями. Також важливим є те, яким чином в алгоритмі сортування реалізована обробка відсортованих даних.

• Складність реалізації алгоритму сортування визначає затрати на програмну реалізацію алгоритму, наприклад метод сортування “merge sort” є значно складнішим у реалізації, ніж метод “бульбашки”.

• Об'єм використовуваних ресурсів вказує на необхідний для роботи алгоритму обсяг оперативної чи дискової пам'яті, час виконання сортування, тощо. В процесі роботи деяких алгоритмів слід виділяти додатковий об'єм пам'яті для збереження тимчасових даних. При збільшенні кількості елементів, які необхідно відсортувати збільшується як обсяг необхідної пам'яті так і зростає час сортування, причому різниця у швидкодії алгоритмів стає більш помітною.

• Швидкодія роботи алгоритму може обчислюватися кількістю ітерацій, або часом необхідними для сортування.

Сортування може здійснюватися таким чином:

• вставкою — сортування відбувається шляхом поелементного порівняння поточного операнда із рештою, зсуву та записом його у вакантну позицію;

• об'єднанням — процес сортування полягає у розбитті масиву на окремі блоки в межах яких здійснюється сортування та подальшого об'єднання відсортованих елементів у один вихідний масив;

• перестановкою — сортування відбувається шляхом порівняння двох сусідніх

операндів, збереження їх в пам'яті та обміну позиціями;
• вибіркою — сортування полягає у пошуку найменшого значень із заданого діапазону в масиві та його запису на початку діапазону.

Сортування може бути стійким, або не стійким. Якщо сортування є стійким, після його завершення зберігається порядок розташування елементів з однаковим ключем. Стійке сортування застосовується в тому випадку, коли необхідно відсортувати масиви даних за кількома параметрами із збереженням початкового розташування елементів вхідних даних. Прикладом застосування такого типу сортування може бути впорядкування записів таблиці, що зберігає прізвища, імена та телефони людей. Цей тип сортування забезпечує більшу швидкодію за рахунок відмови від виконання додаткових операцій перестановки елементів в масиві чи таблиці, які вже відсортовані. Стійке сортування забезпечують алгоритми “insert sort”, “merge sort”, “bubble sort”.

Таблиця 3.1

Характеристика алгоритмів сортування даних

Назва методу	Метод сортування	Середня кількість ітерацій, n -довжина масиву	Характеристика
Bubble sort	обмін	n^2	Проста реалізація
Gnome sort	обмін	n^2	Проста реалізація
Insert sort	вставка	n^2	Проста реалізація
Merge sort	об'єднання	n•log(n)	Проста реалізація
Quick sort	обмін	n•log(n)	Складна реалізація

Алгоритми сортування

Сортування простим обміном (“bubble sort”) є простим алгоритмом сортування невеликих масивів даних. В основному він використовується для ознайомлення та вивчення простих алгоритмів сортування. Суть алгоритму полягає у виконанні порівнянь двох сусідніх елементів масиву, обміну їхніми позиціями за необхідності та переходом на нову позицію. Робота алгоритму організовується за допомогою двох циклів. Зовнішній цикл призначений для сортування всіх елементів масиву, а внутрішній для сортування одного елементу в масиві. Під час виконання внутрішнього циклу здійснюється операція порівняння двох сусідніх елементів та обміну їхніми позиціями в

результаті чого поточне значення рухається у праву частину масиву. Таке переміщення нагадує підняття бульбашки на поверхню води. Для здійснення повного сортування необхідно виконати n^2 ітерацій.

Алгоритм сортування “gnome sort”

Алгоритм сортування запропоновано Діком Груном (Dick Grune). Він відноситься до простих методів сортування даних. Сортування, згідно цього алгоритму, є подібним до роботи садового гнома, який здійснює перестановку садових горщечків з переходом до наступного. Саме сортування відбувається шляхом порівняння значень двох сусідніх елементів масиву та **обміну** значеннями на цих позиціях.

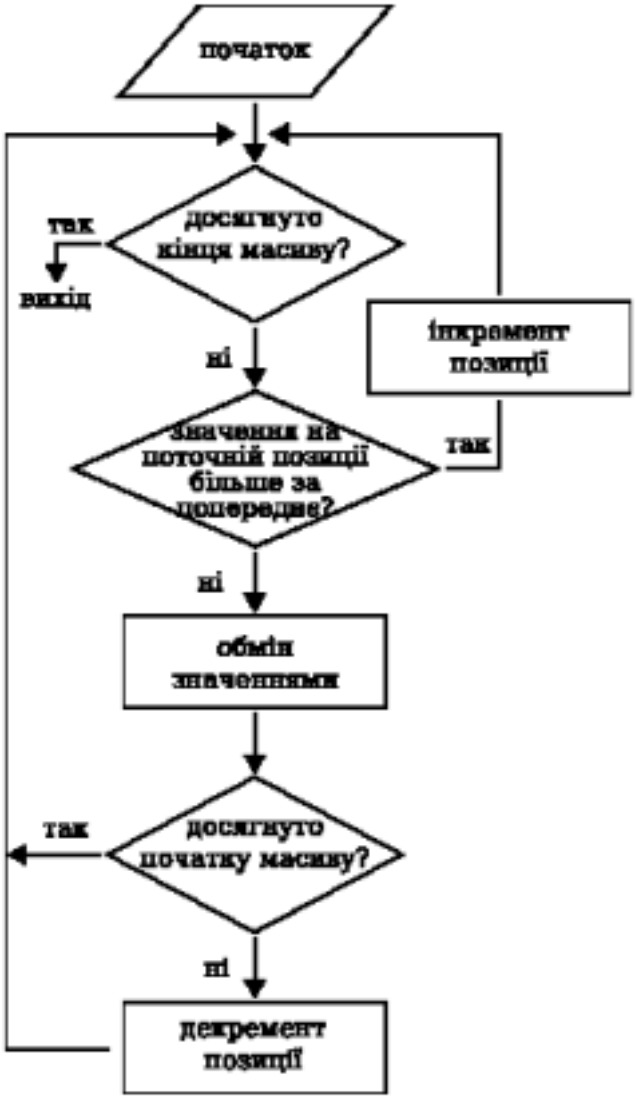

Рис. 3.1. Блок-схема роботи алгоритму сортування “gnome sort”

Якщо елемент, який знаходиться на позиції **n** є меншим за елемент на позиції **n-1**, відбувається обмін їхніми позиціями (значеннями) та перехід на одну позицію назад. Далі відбувається повторне порівняння і за необхідності здійснюється обмін позиціями.

Такий процес повторяється до того моменту поки значення на поточній позиції буде меншим за значення на попередній, або повернення до початку масиву. В протилежному випадку, коли значення елементу на поточній позиції є більшим за значення на попередній позиції, відбувається перехід на одну позицію вперед. Алгоритм сортування “gnome sort” за своєю роботою **є подібним до** “bubble sort” в якому сортування здійснюється на основі серії обмінів. Проте це два різні алгоритми сортування! Роботу алгоритму доцільно реалізувати у вигляді циклу, в якому значення лічильника циклу (поточна позиція сортування) змінюється в процесі сортування залежно від значення досліджуваних елементів.

Приклад сортування “gnome sort” послідовності випадкових чисел 5 3 1:
етап 01: початкова послідовність

5 3 1

етап 02: розпочинаємо сортування з першої позиції, порівнюємо з попереднім значенням; необхідно відсортувати 5 > 3 (здійснюємо обмін значеннями)

5 3 1

етап 03: переходимо на одну позицію назад; порівняти з попереднім значенням неможливо (воно відсутнє)

3 5 1

етап 04: переходимо на одну позицію вперед, порівнюємо з попереднім значенням; сортування непотрібне 3 < 5

3 5 1

етап 05: переходимо на одну позицію вперед; порівнюємо з попереднім значенням; необхідно відсортувати 5 > 1 (здійснюємо обмін значеннями)

3 5 1

етап 06: переходимо на одну позицію назад; порівнюємо з попереднім значенням; необхідно відсортувати 3 > 1 (здійснюємо обмін значеннями)

3 1 5

етап 07: переходимо на одну позицію назад; порівняти з попереднім значенням неможливо (воно відсутнє)

1 3 5

етап 08: переходимо на одну позицію вперед; порівнюємо з попереднім значенням; сортування непотрібне 1 < 3

1 3 5

етап 09: переходимо на одну позицію вперед; порівнюємо з попереднім значенням; сортування непотрібне 3 < 5

1 3 5

етап 10: переходимо на одну позицію вперед; оскільки досягнуто кінця масиву робота алгоритму завершується; відсортована послідовність є такою:

1 3 5

Процес сортування “gnome sort”, один з прикладів якого описаний вище, використовує лише операції порівняння та обмін значеннями.

Алгоритм сортування “insert sort”

На відміну від алгоритму “gnome sort” сортування даних в “insert sort”, здійснюється шляхом виділення поточного елементу масиву, виконання серії **переміщень** та його **вставки** у вакантну позицію.

Сортування даних згідно алгоритму “insert sort” відбувається у кілька етапів. Спочатку вибирається елемент масиву на активній позиції. Це значення зберігається у тимчасовій змінній та порівнюється із попереднім елементом. Якщо воно є меншим за нього відбувається перезапис попереднього елементу у поточну позицію та декремент значення позиції. Ці операції повторюються доти, поки поточне значення буде меншим за попереднє, або досягнуто початку масиву.

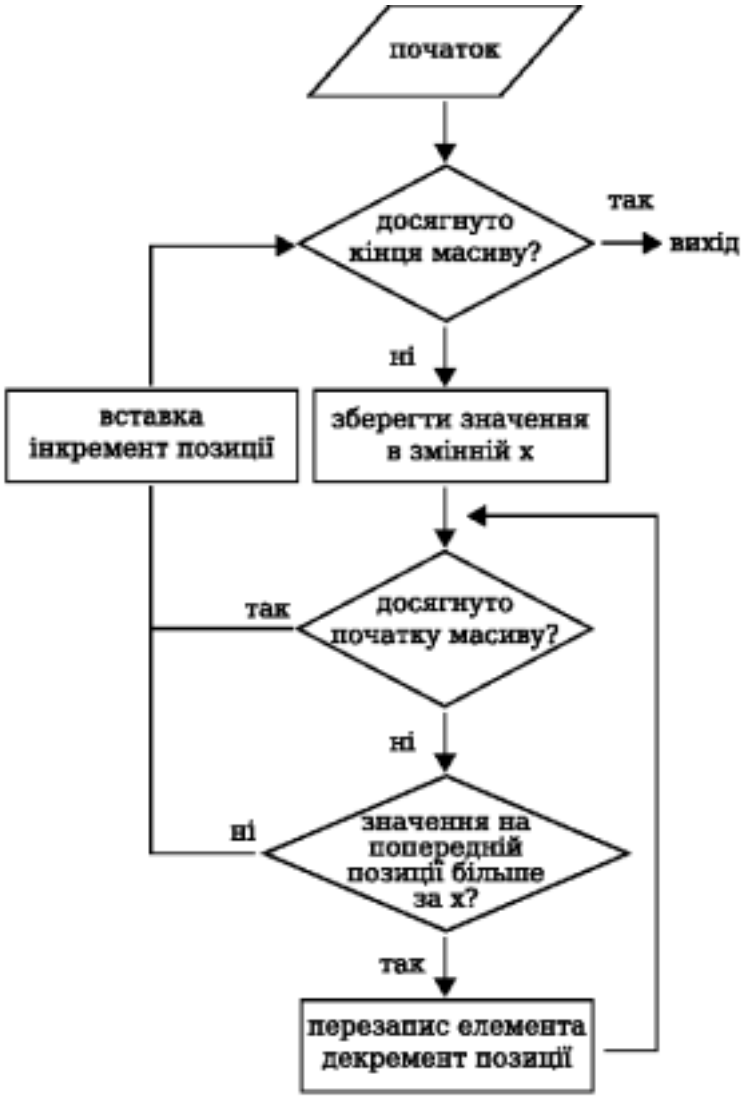

Рис. 3.2. Блок-схема роботи алгоритму сортування “insert sort”

Після завершення цих операцій у вакантну позицію, утворену внаслідок зсуву елементів масиву вправо, записується значення, що зберігалось у тимчасовій змінній. Якщо поточне значення є більшим за попереднє відбувається інкремент активної позиції.

Роботу алгоритму слід реалізувати за допомогою двох циклів. Зовнішній цикл повинен забезпечити сортування всіх елементів масиву, а внутрішній — сортування одного елементу.

Алгоритми “gnome sort” та “insert sort” є близькими у підході щодо сортування даних, проте використовують **різні методи** сортування!

Приклад сортування “insert sort” послідовності випадкових чисел 5 3 1:

етап 01: початкова послідовність

5 3 1 x = ?

етап 02: розпочинаємо сортування з першої позиції; записуємо значення активної позиції в тимчасову змінну;

5 3 1 x = 3

етап 03: порівнюємо значення тимчасової змінної з попереднім; необхідно відсортувати 5 > 3 (здійснюємо перезапис 5 заміняє 3)

5 5 1 x = 3

етап 04: переходимо на одну позицію назад; порівняти з попереднім значенням неможливо (воно відсутнє); робимо вставку (заміняємо значення на активній позиції на x)

3 5 1 x = 3

етап 05: переходимо на одну позицію вперед; записуємо значення активної позиції в тимчасову змінну; порівнюємо з попереднім значенням; сортування непотрібне 3 < 5

3 5 1 x = 5

етап 06: переходимо на одну позицію вперед; записуємо значення активної позиції в тимчасову змінну;

3 5 1 x = 1

етап 07: порівнюємо з попереднім значенням; необхідно відсортувати 5 > 1 (здійснюємо перезапис 5 заміняє 1)

3 5 5 x = 1

етап 08: переходимо на одну позицію назад;

3 5 5 x = 1

етап 09: порівнюємо з попереднім значенням; необхідно відсортувати 3 > 1 (здійснюємо перезапис 3 заміняє 5)

3 3 5 x = 1

етап 10: переходимо на одну позицію назад;

3 3 5 x = 1

етап 11: порівняти з попереднім значенням неможливо (воно відсутнє); робимо вставку (заміняємо значення на активній позиції на x)

1 3 5 x = 1

етап 12: переходимо на одну позицію вперед; записуємо значення активної позиції в тимчасову змінну; порівнюємо з попереднім значенням; сортування непотрібне 1 < 3

1 3 5 x = 3

етап 13: переходимо на одну позицію вперед; записуємо значення активної позиції в тимчасову змінну; порівнюємо з попереднім значенням; сортування непотрібне 3 < 5

1 3 5 x = 5

етап 14: переходимо на одну позицію вперед; оскільки досягнуто кінця масиву робота алгоритму завершується; відсортована послідовність є такою:

1 3 5

Процес сортування “insert sort”, один з прикладів якого описаний вище, використовує операції порівняння, перезапис та вставка.

Порядок роботи

1. Запустити середовище розробки програм на мові C/C++ (BorlandC, GCC, MinGW, Dev-C++, Visual Studio, тощо).
2. Вибрати із залікової книжки дві останні цифри **m** (передостання) та **n** (остання). Якщо будь-яка із цифр рівна нулеві замінити її на 1.
3. Для виконання завдання попередньо заповнити масив g[] із 20 елементів випадковими значеннями з довільним законом розподілу.
4. Скласти програму для сортування елементів масиву з використанням алгоритму “gnome sort” та “insert sort”.
5. Пояснити хід сортування двома алгоритмами.

Контрольні запитання

1. Що таке сортування?
2. Які існують вимоги до алгоритмів сортування?
3. В чому полягає надійність сортування?
4. Які існують методи сортування?
5. Які операції використовуються під час сортування?
6. Що таке стійкий алгоритм сортування?
7. Які існують стійкі алгоритми сортування?
8. Яким чином здійснюється сортування “bubble sort”?
9. Яким чином здійснюється сортування “gnome sort”?
10. Яким чином здійснюється сортування “insert sort”?

Зміст звіту

1. Титульний лист
2. Тема та мета роботи
3. Короткі теоретичні відомості
4. **Результати виконаної роботи**
5. **Висновок**

ЛІТЕРАТУРА

1. Березин Б.И., Березин С.Б. Программирование на C и C++ - М.: ДИАЛОГ-МИФИ, 2001. - 288 с.
2. Керниган Б., Ритчи Д.. Язык программирования C, 2-е издание - М.: Вильямс — 2009. - 292 с.
3. Лафоре Л. Объектно-ориентирование программирование в C++, 4-е издание — М.: Питер, 2004. - 923 с.
4. Папас К., Мюррей У. Программирование на C и C++ - К.: Издательская група BHV, 2000. - 320 с.
5. Шилдт Г. Справочник программиста C/C++, 3-е изд.: Пер. с англ. - М. Издательский дом “Вильямс”, 2003. - 432 с.
6. Heineman G. Algorithms in a nut shell. O'Reilly Media, 2009. - 341 p.
7. Sedgewick R. Algorithms in C. Addison-wesley publishing. 1990. - 657 p.
8. http://en.wikipedia.org/wiki/Gnome_sort
9. http://en.wikipedia.org/wiki/Insertion_sort