ЛЕКЦІЯ 8.

ЛІНІЙНІ СТРУКТУРИ ДАНИХ.

8.1. Лінійні списки . Основні визначення та поняття

Раніше розглядались структури, в яких зв'язок між елементами заданий неявно. Однак в класі лінійних структур даних існують значно складніші зв'язні структури, в яких функціональний зв'язок між його елементами заданий явно. До таких структур належать спискові. Розглянемо найпростіші з таких структур, а саме, лінійні списки.

Список, що відображає відношення сусідства між елементами, називають**лінійним.** Будь-який інший список вважається нелінійним. Тип списків визначається типом зв'язків між його елементами. За кількістю зв'язків розрізняють списки одно- і багатозв'язні, а за типом функції зв'язку - лінійні і нелінійні.

Основний принцип спискової структури даних полягає в тому, що логічний порядок слідування елементів задається сукупністю посилань або вказівників. У послідовній пам'яті дані розміщуються в послідовних одиницях пам'яті і цим визначається порядок їх слідування.

Отже, поняттю "список" можна дати таке пояснення: організацію зберігання даних, при якій логічний порядок слідування даних визначається посиланнями або вказівниками, називають **списковою організацією пам'яті**, а дані, що зберігаються таким чином, - **списком**. Спискова організація пам'яті має також назву зчеплення.

Елемент списку складається як мінімум з двох полів - безпосереднього значення елемента даних і вказівника на наступний елемент списку. *У* деяких випадках значення елементів даних можуть зберігатися окремо, тоді у полі "значення" елемента списку може знаходитися вказівник на місцезнаходження цього елемента даних. Вказівники елементів списку називають також адресами зв'язку, або зв'язками.

Однією з основних властивостей списків є те, що елементи розміщуються в пам'яті довільним способом, а не в суміжних одиницях пам'яті, як у векторів.

Кожний список має свій **заголовок**, у якому зберігається посилання на перший елемент списку. Заголовок списку називають також **іменем списку**. Всі інші елементи досягаються шляхом проходження списку за допомогою вказівників.

Існує два способи зображення списків : 1) графічний (рис.8.1,а); 2) дужковий (рис.8.1,б).

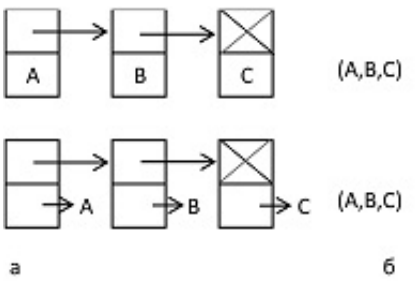

Рис.8.1. Приклади зображення списків: а) графічний; б) дужковий

При графічному способі список зображується у вигляді ланцюга, кожна ділянка якого складається з двох полів - довідки і тіла. Дужковий вираз для зображення списку зручний тоді, коли списки складаються з різних типів даних і зміна одного списку не впливає на другий.

Наприклад, на рис.8.1,а зображено список, у якого в полі тіла елемента записано безпосередньо значення елемента даних, а на рис.8.1,б - вказівник на місцезнаходження елемента даних. Дужковий вираз у обох випадках залишається однаковим і не відображає різного вмістимого поля тіла елементів списку.

При обробці списків найчастіше виконуються такі дії:

1) доступ до *к* -го елемента списку з метою аналізу і заміни його полів;

2) включення нового елемента безпосередньо перед заданим;

3) виключення заданого елемента;

4) об'єднання декількох списків в один;

5) розбиття списку на два або більше списки;

6) копіювання списку;

7) визначення кількості елементів у списку;

8) знаходження елемента за заданими властивостями;

9) пересортування або впорядкування елементів списку у висхідному або низхідному порядку.

8.2. Однонаправлені списки

Найбільш природним і простим типом списку є **однонаправлений**, призначений для того, щоб проглядати його в одному напрямі - від початку до кінця. Однонаправлений список найчастіше зображують у вигляді ланцюга, тому його називають також ланцюговим, або лінійним однозв'язним (рис.8.2).

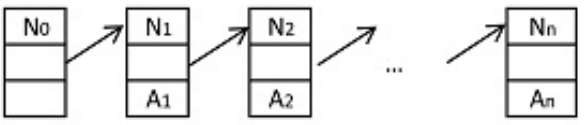

Рис.8.2. Приклад однонаправленого списку

Довідка кожної ділянки такого списку складається з двох значень. Першим є довжина N_i *i*-ї ділянки, яка, в свою чергу, складається з довжини запису і довжини довідки. Другим значенням довідки є посилання на початок наступної ділянки. Далі розміщується тіло ділянки або безпосередньо запис. **Заголовна** ділянка складається з трьох полів: довжини заголовної ділянки; посилання на початок першої ділянки; посилання на початок вільного місця. Перші два значення утворюють довідку заголовної ділянки, а посилання на вільне місце являє собою тіло цієї ділянки. У довідці останньої ділянки поле вказівника порожнє. У загальному випадку всі записи в однонаправленому списку можуть мати різну довжину.

Однонаправлений список можна відобразити в одномірний масивнаступним способом. Заголовний запис буде займати елементи масиву *S[o] , S[1], S[2]*. Далі розміщуються інші записи. Якщо індекс ділянки, що містить запис *X* , має значення *i* , то в елементі *S[i]* знаходяться довжина цієї ділянки, в елементі *S[i+1]* - індекс ділянки з наступним записом списку (або нуль, якщо запис *X* останній). Починаючи з елемента *S[i+2]* знаходиться сам запис (тіло ділянки).

Схема процедури **включення** нового запису *Z* після ділянки з індексом *i*показана на рис. 8.3. При цьому спочатку формується нова ділянка з записом *Z*, яка розміщується на вільному місці

пам‘яті, а потім коректується зв‘язок *i* -ї ділянки.

Рис.8.3. Схема процедури включення елемента в ланцюговий список

Процедура **виключення** елемента з такого списку також зводиться до корекції зв‘язків між ділянками (рис.8.4). Припустимо, що потрібно виключити зі списку ділянку, яка розміщується після ділянки з індексом *i* . Отже, індекс ділянки, що виключається, розміщений в *S[i+1]*. Значення *S[i+1]* заміняється на значення, яке зберігалось у довідці ділянки, що виключається.

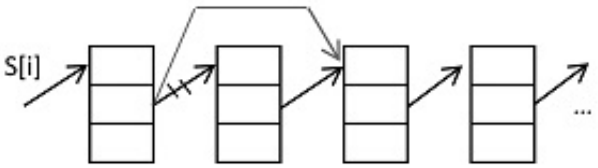

Рис. 8.4. Схема процедури виключення елемента

з однонаправленого списку

Коли зі списку виключаються записи, виникає питання про використання того місця пам‘яті, де зберігалися виключені записи. Якщо всі записи мають однакову довжину, нові записи можна розміщувати на цих місцях, для чого з виключених ділянок утворюється **список вільних ділянок**. Якщо ж ділянки мають різну довжину, задача значно, ускладнюється. Один із методів її розв‘язку полягає у тому, щоб час від часу утворювалась копія списку, куди входили б тільки дійсні записи.

Всякий список супроводжується списком вільної пам‘яті, який готується обслуговуючими програмами. Такий список має свій власний вказівник, який містить адресу першого вільного елемента пам‘яті, а також число таких елементів. Перший вільний елемент містить адресу другого елемента і т.д. Але побудова такого списку вільної пам‘яті значно ускладнюється, якщо записи елементів мають різну довжину. Тоді до довідки ділянки списку вільної пам‘яті необхідно приєднувати також довжину запису. Процедуру, пов‘язану з поверненням вільних квантів пам'яті у список вільної пам'яті, називають "**збиранням сміття**".

8.3. Двонаправлені списки

Двонаправлені списки це такі списки, які дозволяють рухатися вперед і назад при перегляді його елементів. Вони зображуються у вигляді ланцюга, кожна ділянка якого містить вказівники на наступний і попередній елементи.

На рис.8.5 зображено структуру заголовної (рис.8.5,а) і внутрішньої (рис.8.5,б) ділянок, а також самого двонаправленого списку (рис.8.5,в). Кожна ділянка такого списку складається з чотирьох значень (рис.8.5,б), а саме: значення самого елемента даних і довідки стандартного вигляду з трьох значень - довжини ділянки, вказівників на наступну і попередню ділянки. В загальному

випадку довжина кожної ділянки може бути різною. Заголовна ділянка також складається з чотирьох значень, але її довжина відома і рівна чотирьом, оскільки записом її елемента є вказівник, а всі вказівники мають однакову довжину (рис.8.5,а).

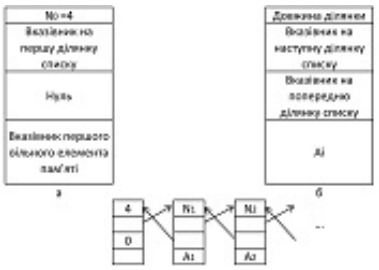

Рис. 8.5. Зображення двонаправленого списку і його ділянок

Очевидно, що операції включення-виключення запису в даному випадку аналогічні до попередніх і також зводяться до коректування зв‘язків між ділянками. Однак операція включення запису в двонаправлений список буде значно складнішою від операції виключення запису з нього, оскільки в першій операції необхідно прочитувати дві ділянки, а в другій - тільки ту, яка виключається. Попередній запис потрібний для коректування зв‘язку на наступний елемент, але цей запис не треба читати, знайдемо його у ділянці, яка виключається. Тому можна записати туди нову адресу без читання самої ділянки.

Двонаправлений список також відображується в одномірний масив, починаючи зі стандартної заголовної ділянки, яка має в масиві індекси 0, 1, 2, 3.

8.4. Циклічні списки

Циклічні, або кільцеві, списки - найпоширеніша різновидність двонаправлених списків. У такому списку остання ділянка посилається на заголовну як на наступну, а ця, в свою чергу, посилається на останню ділянку як на попередню. Така організація спрощує процедуру пошуку або перебору записів з довільного місця з автоматичним переходом від кінця на початок, і навпаки. Бувають також однозв‘язні кільцеві списки, в яких остання ділянка посилається на заголовну як на наступну. На рис.8.6 зображено циклічний двозв‘язаний список.

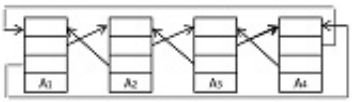

Рис.8.6. Циклічний список

Операція **включення** в кільцевий двозв‘язаний список подібна до попередніх. Наприклад, нехай потрібно встановити ділянку з новим записом після ділянки з індексом *i*. Нова ділянка розміщується на вільному місці пам‘яті і включається у список шляхом коректування вказівників в ділянці з індексом *i* , а також у тій ділянці, яка раніше була після неї.

Операція **виключення** ділянки із кільцевого двозв‘язного списку також зводиться до коректування двох посилань - у "сусіда ліворуч" змінюється вказівник на наступну ділянку, а у "сусіда праворуч" - вказівник на попередню ділянку.

Циклічний двозв‘язний список можна відобразити у одномірний масивааналогічно лінійному списку. У такому випадку процедури включення-виключення елементів зводяться до обчислення індексів вказівників, як і в лінійному списку, і заміни вмістимого елементів масиву з цими індексами. Якщо тіло елемента списку являв собою рядок символів або бітів, а вказівники -

цілі числа, то не в кожній мові програмування таке відображення можливе. Тому, наприклад, простіше відображувати списки у два масиви - один для тіла елемента, другий - для довідкової частини.

Частіше, однак, елементи списку не розташовують у якому-небудь масиві, а просто розміщають кожний окремо в оперативній пам'яті, виділеній даній задачі. Зазвичай елементи списку розміщаються в динамічній пам'яті. У якості вказівників у цьому випадку використовують адреси елементів в оперативній пам'яті.

Голова списку зберігає вказівник на перший і останній елементи списку. Оскільки список зациклений у кільце, то наступним за головою списку буде його перший елемент, а попереднім - останній елемент. Голова списку зберігає тільки вказівник й не зберігає ніякого елемента. Це як би порожня комірка, у якій не можна нічого покласти і яка використовується тільки для того, щоб зберігати адреси наступного й попереднього елементів списку, тобто першого й останнього. Коли список порожній, голова списку зациклена сама на себе.

Перевага вказівникової реалізації списку полягає в тому, що процедури додавання й видалення елементів не приводять до масових операцій.