Розділ 1

Структура даних

1.1 Поняття структури даних

Сучасна ЕОМ задумувалася та будувалась як пристрій, що автоматизує та полегшує складні математичні розрахунки, які вимагають значних затрат часу. На сьогоднішній день на перше місце виступає можливість зберігати значні об'єми даних із можливістю швидкого доступу до них. Можливість же виконувати обчислення при цьому відступила на другий план.

Ми говоримо про дані як про деяке абстрактне представлення реального світу: деякі несуттєві характеристики та властивості об'єктів при цьому ігноруються. Розв'язок будь-якої задачі навіть без використання ЕОМ вимагає певного рівня абстрагування - визначення множини даних, що характеризують реальну ситуацію.

На етапі розв'язку задачі та розробки відповідного програмного забезпечення необхідно визначити структуру та формат використовуваних даних. При цьому під структурою даних розуміють множину елементів даних та взаємозв'язків між ними. Незалежно від суті та складності будь-які дані в оперативній пам'яті комп'ютера задаються у вигляді послідовності двійкових розрядів (бітів), а їх значеннями є відповідні двійкові числа. Бітові послідовності є слабо структурованими та незручними для практичного використання. На практиці, за звичай, використовують більш складну організацію структури даних.

Будь-які достатньо великі програми проектуються шляхом декомпозиції задачі – виділення в задачі деяких структур і їхніх абстракцій. Абстрагування від проблеми передбачає ігнорування ряду деталей для того, щоб звести задачу до більш простої. Задачі абстрагування і наступна декомпозиція типові для процесу створення програм: декомпозиція використовується для поділу програм на компоненти; абстрагування ж передбачає продуманий вибір

компонентів для цієї задачі.

Структурний підхід до даних і алгоритмів дає можливість структурування складної програми. Розробляти сучасну програмну методом „все відразу" неможливо, вона повинна бути представлена в вигляді деякої структури – складових частин і зв'язків між ними. Правильне структурування дає можливість на кожному етапі розробки зосередити увагу розробника на одній оглядовій її частині або доручити реалізацію різних її частин різним виконавцям.

1.1.1 Концепція типу даних

Структури даних і алгоритми є тими матеріалами, з яких будуються програми. Більше того, сам комп'ютер складається з структур даних і алгоритмів. Вбудовані структури даних представлені регістрами й словами пам'яті, де зберігаються бінарні величини. Закладені в конструкцію апаратури алгоритми – це реалізація в електронних логічних схемах жорстких правил, за якими поміщені в пам'ять дані інтерпретуються як команди, що підлягають виконанню. Тому в основі роботи будь-якого комп'ютера лежить вміння оперувати лише з одним видом даних – з окремими бітами. Працює ж з цими даними комп'ютер тільки у відповідності з незмінним набором алгоритмів, які визначаються системою команд центрального процесора.

Задачі, які вирішуються за допомогою комп'ютера, рідко представляються мовою бітів. Як правило, дані мають форму чисел, літер, текстів, символів і більш складних структур типу послідовностей, списків і дерев.

Структура даних відноситься за своєю суттю до „просторових" понять: її можна звести до схеми організації інформації в пам'яті комп'ютера. Алгоритм ж є відповідним процедурним елементом в структурі програми – він служить рецептом розрахунку.

Структури даних, які застосовуються в алгоритмах, можуть бути досить складними. Вибір правильного представлення даних часто служить ключем до вдалого програмування і може в більшій мірі впливати на продуктивність програми, ніж деталі реалізації використовуваного алгоритму. Але, мабуть, ніколи не появиться загальна теорія вибору структур даних, у кожному конкретному випадку потрібно підходити до цього творчо.

Незалежно від змісту і складності будь-які дані в пам'яті комп'ютера представляються послідовністю бінарних розрядів, а їх значеннями є відповідні бінарні числа. Дані, які розглядаються у вигляді послідовності бітів, мають дуже просту організацію, тобто є слабо структурованими. Більш крупні й змістовніші, ніж біт, „будівельні блоки" для організації довільних даних отримуються на основі поняття „структури даних".

Поняття „фізична структура даних" відображає спосіб фізичного представлення даних в пам'яті машини і називається ще структурою зберігання. Розгляд структури даних без врахування її представлення в машинній пам'яті називається абстрактною або логічною структурою. В загальному випадку між логічною і відповідною їй фізичною структурами існує відмінність, міра якої залежить від самої структури і особливостей того середовища, в якому вона повинна бути відображена.

Розгляд структур даних утруднений досить незручним змішуванням аб-

стрaктних властивостей структур даних і проблемами представлення, які зв'язані з комп'ютером і машинною мовою. Важливо чітко розрізняти ці проблеми за допомогою багаторівневого опису. Будь-яка структура даних може описуватися, таким чином, на трьох різних рівнях:

- **функціональна специфікація** – вказує для деякого класу імен операції, які дозволені з цими іменами, і властивості цих операцій;
- **логічний опис** – задає декомпозицію об'єктів на більш елементарні об'єкти і декомпозицію відповідних операцій на більш елементарні операції;
- **фізичне представлення** – дає метод розміщення в пам'яті комп'ютера тих величин, які складають структуру, і відношення між ними, а також спосіб кодування операцій на мові програмування.

Одній і тій же функціональній специфікації можуть відповідати декілька логічних описів, які в свою чергу можуть реалізовуватися декількома фізичними представленнями. Проте, потрібно мати впевненість, що декомпозиція кожного нового рівня достатньо добре відображає декомпозицію безпосередньо вищого рівня.

Найпростіші структури даних, реалізовані мовою програмування, називають також стандартними типами даних . Багато мов програмування дозволяють на основі стандартних типів будувати типи даних, визначені програмістом (користувачем).

1.1.2 Класифікація структур даних

Із поняттям структури данных тісно пов'язане власне поняття типу даних. Розрізняють фізичну та логічну структуру даних. Фізична структура, на відміну від логічної, визначає спосіб представлення даних в пам'яті комп'ютера.

Розрізняють прості структури данных (типи) та складні (інтегровані). Прості структури не можуть бути поділені на складові частини, більші ніж біти. З точки зору фізичної структури для простого типу даних чітко визначений його розмір та спосіб розміщення в оперативній пам'яті комп'ютера. З точки зору логічної структури прості дані є неподільними одиницями.

Інтегровані структури даних включають у себе інші структури даних — прості чи інтегровані. Між окремими елементами структури можуть існувати явно задані зв'язки (не обов'язково). В залежності від цього розрізняють: незв'язані структури (вектори, матриці, рядки, стеки, черги) та зя'язані структури (звязані списки).

За ознакою можливості зміни розміру розрізняють структури статичні, напівстатичні та динамічні. Під зміною розуміють зміну числа елементів структури або зв'язків між її елементами. Класифікація структур даних за ознакою зміни приведена на рис.1.

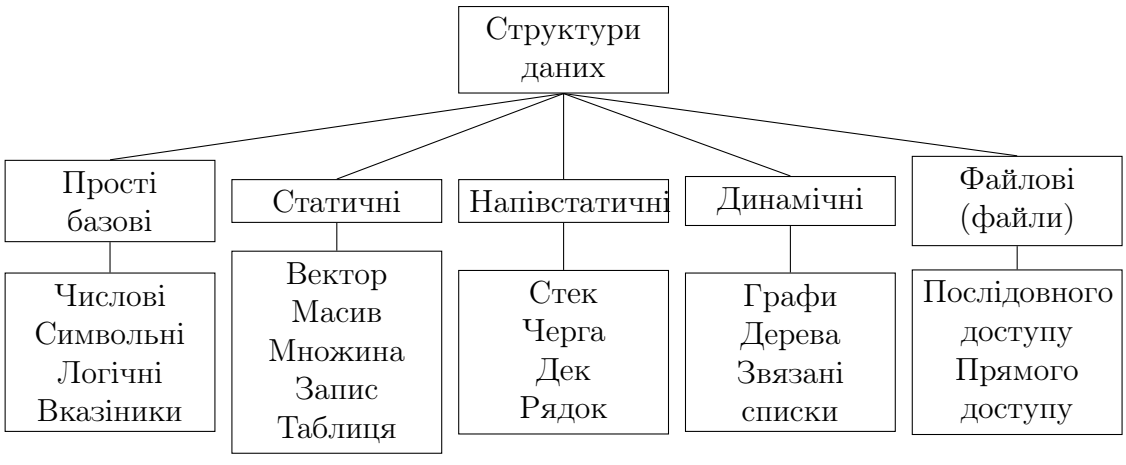

Рис. 1.1: Класифікація структур даних

За ознакою впорядкованості елементів структури можна поділити на лінійні та нелінійні. Прикладом нелінійних структур можуть бути багатозв'язні списки, дерева, графи.

Лінійні структури, в свою чергу, поділяються на структури із послідовним розподілом (вектори, рядки, масиви, стеки, черги) та структури із довільними розподілом (однозв'язні, двохзв'язні списки) за характером розподілу елементів в оперативній пам'яті.

Задання типу даних однозначно визначає:

- розмір пам'яті, який відводиться для збереження даної структури та спосіб її розміщення:
- діапазон допустимих значень, які може приймати структура;
- операції, які можна виконати над даного типу даними.

Прості структури даних є основою для побудови більш складних структур. Їх також називають примітивними чи базовими структурами (типами даних). До них відносять: числові, бітові, логічні, символьні, перераховувані, інтервальні, вказівники. Структура простих типів даних для мови C++ приведена на рис.2 (в інших мовах програмування набір та розмір простих типів можуть відрізнятися від приведеного).

Размер каждого типа данных указан на рисунке в байтах через запятую от названия типа. Как уже было сказано, разные типы данных имеют различный формат представления их в машинной памяти. На рис. 3.3—3.5 приведены примеры форматов числовых типов данных. На рис. 3.4 S обозначает знаковый разряд числа (если S=0, то число положительное, если 5=1— число отрицательное). Формат для представления чисел с плавающей точкой, приведенный на рис. 3.5, а, содержит поля мантиссы, порядка и знаков мантиссы и порядка фиксированной длины. Однако чаще вместо порядка используется характеристика, полученная