Cấu trúc dữ liệu và giải thuật

Chương 2: Các cấu trúc tuyến tính

Phần 1: Cấu trúc Mảng

Các nội dung chính

1. Cấu trúc mảng
   - Mô tả
   - Cấu trúc lưu trữ tuần tự
   - Cài đặt mảng bằng cấu trúc lưu trữ tuần tự
   - Hàm địa chỉ
2. Cấu trúc danh sách
   - Mô tả
   - Cấu trúc vào sau ra trước (LIFO) (Stack-Ngăn xếp)
   - Cấu trúc vào trước ra trước (FIFO) (Queue-Hàng đợi)
   - Một số ứng dụng của ngăn xếp và hàng đợi

1. Cấu trúc mảng

- Mô tả cấu trúc
  - *Mảng (*Array) là một tập cố định các phần tử và cùng kiểu dữ liệu.
  - Tính chất đặc trưng
    - Số chiều: số chiều của mảng tương ứng với số chiều của thông tin cần được biểu diễn. Một mảng bao giờ cũng ít nhất một chiều.
    - Kích thước mỗi chiều: phải là một giá trị cố định. Ta có thể dễ dàng suy ra kích thước của mảng bằng cách lấy tích tất cả các kích thước của tất cả các chiều.
    - Kiểu phần tử mảng: đó là kiểu dữ liệu cho mỗi phần tử của mảng.
  - Kiểu mảng có thể được khái quát bằng khai báo như sau:
    - **ARRAY** : <*name*>[*dimension*, *len 1*, *len 2*,..., *len n*] **OF** *datatype*;
  - Khi đó, kích thước của mảng *name* kí hiệu *LEN(name)* được tính bằng công thức:
    - *LEN(name) = len 1 x len 2 x ...* = Π *(len i)* (với i=1,2,..,n)

1. Cấu trúc mảng

- Mô tả - Ví dụ:
  - Khai báo mảng 1 chiều:
    - ARRAY: vector [1, N] OF integer ;
  - Khai báo mảng hai chiều:
    - ARRAY: matran[2, M, N] OF integer; hay tương đương
    - ARRAY: matran[1, M] OF vector;
  - Khai báo mảng N chiều:
    - ARRAY : a[N, L_1, L_2,..., Ln] OF integer; *(2.1)*
  - Từ việc khai báo mảng như trên ta có thể dễ dàng suy ra mảng hai chiều là mảng một chiều của các mảng một chiều, mảng ba chiều là mảng 1 chiều của các mảng 2 chiều,..., mảng N chiều là mảng 1 chiều của các mảng N-1 chiều.

1. Cấu trúc mảng

- Mô tả - Ví dụ:
  - Chúng ta có thể mô tả hình thức như sau:
    - ARRAY: $a_n[N, L_1, L_2,..., Ln]$ OF datatype ; $\Leftrightarrow$
    - ARRAY: $a_{n-1}[N-1, L_2,..., Ln]$ OF datatype ; AND
      ARRAY: $a_n[1, L_1]$ OF a_{n-1} ; *(2.2)*
  - Trong C/C++, các mảng trên được khai báo như sau:
    - int vector [0..N-1] ;
    - int matran [0..M-1][0..N-1] ; hay
    - vector matran [0..M-1];
  - Lưu ý trong C/C++ quy ước:
    - Kích thước mỗi chiều = chỉ số trên + 1

1. Cấu trúc mảng

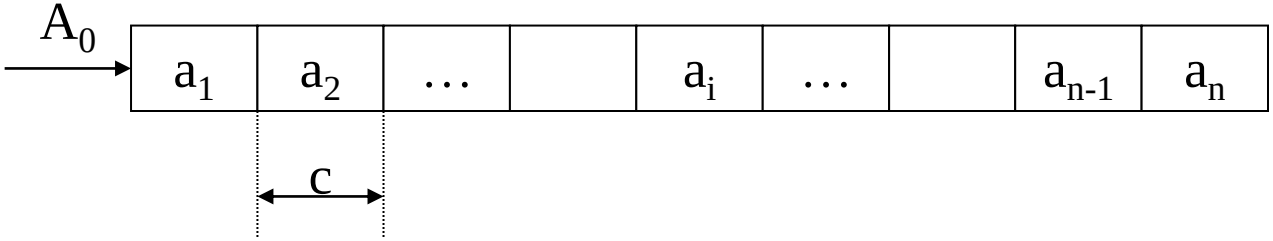
- Các thao tác cơ bản
  - Thao tác khởi tạo: thao tác khởi tạo cấu trúc, xác định các đặc trưng của cấu trúc này. Thao tác này luôn được tiến hành trước tiên. Trong các ngôn ngữ lập trình, thao tác này tương ứng với việc khai báo kiểu dữ liệu mới.
  - Thao tác truy nhập vào các phần tử của mảng: truy nhập vào các phần tử của mảng để sử dụng các phần tử này như: lấy giá trị, cập nhật giá trị.
    - Để truy nhập vào một phần tử của mảng, ta dùng một *chỉ số (index)* gắn với phần tử đó.
    - Mỗi phần tử của mảng có một chỉ số duy nhất, có vai trò như địa chỉ của phần tử trong mảng.
    - Nếu mảng có N chiều như được khai báo ở *(2.1)* thì cấu trúc chỉ số của mỗi phần tử như sau: $[i_1,i_2,...,i_n]$, với i_j (j = 1..N) là các số nguyên thoả mãn: $1 \leq i_j \leq L_j$.

Ví dụ về các thao tác

- Khi truy nhập vào một phần tử thứ i của vector (mảng 1 chiều) ta có: *vector[i]*
- Khi truy nhập vào một phần tử ở hàng i, cột j của ma trận (mảng 2 chiều) ta có: *matran[i,j]*.
- Ta sẽ tìm hiểu sâu hơn về ý nghĩa của chỉ số ở phần sau, khi ta học về khái niệm hàm địa chỉ.

Cấu trúc lưu trữ tuần tự

- Mô tả



  - A_0 là địa chỉ bắt đầu của cấu trúc lưu trữ, cũng là địa chỉ của ô nhớ chứa phần tử đầu tiên.
  - Kích thước mỗi ô nhớ là như nhau, là một hằng số cố định được kí hiệu là c (đơn vị tính thường là byte).
  - Hàm địa chỉ:
    - Địa chỉ của ai: Loc (ai) = A_0 + c* (i-1)
    - Hàm địa chỉ: f(i) = c * (i-1) (address function)

Cấu trúc lưu trữ tuần tự

- Đặc điểm
  - Cấu trúc tương đối đơn giản, dễ sử dụng
  - Kích thước luôn cố định. Việc cấp phát vùng nhớ cho CTLT này được thực hiện đúng một lần, và cũng được giải phóng đúng một lần khi CTLT này không cần dùng nữa (như sau khi ra khỏi một thủ tục hay kết thúc chương trình có sử dụng CTLT này).
  - Việc truy nhập vào các phần tử nhanh và đồng đều (truy nhập trực tiếp) do địa chỉ mỗi phần tử có thể tính trực tiếp. Ta sẽ tìm hiểu cách tính này ở phần sau.
  - Vì cấu trúc mảng có kích thước cố định, gồm các phần tử có cùng kiểu dữ liệu, nên nó thường được cài đặt bằng cấu trúc lưu trữ tuần tự.

Cài đặt mảng bằng cấu trúc lưu trữ tuần tự

- Cài đặt mảng một chiều:
  - ARRAY : a_1[1, N] OF *datatype* ;
  - Thứ nhất, xác định các đặc trưng của cấu trúc lưu trữ:
    - Số ô nhớ : bằng N, là số phần tử của mảng, tức là kích thước của mảng.
    - Kích thước mỗi ô nhớ: là một hằng số *c* cố định mà bằng kích thước của kiểu dữ liệu *datatype* của mỗi phần tử của mảng.
    - Cần dành ra một khối nhớ liên tục có kích thước c.N, có địa chỉ đầu tiên là A_0 để lưu trữ cho mảng này.

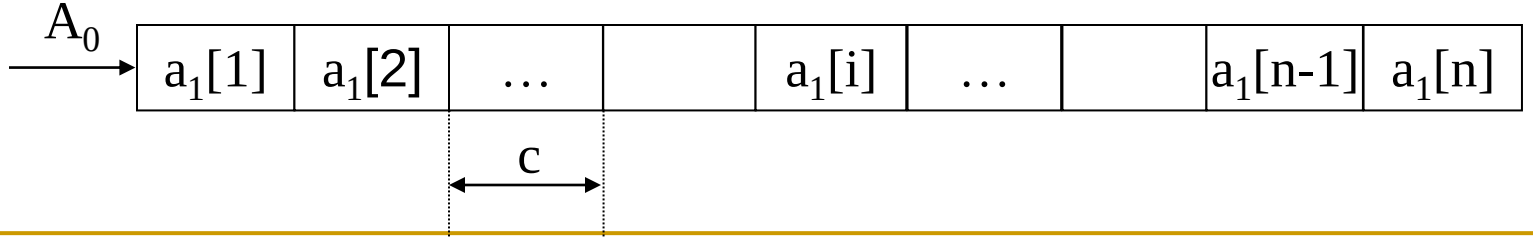
Cài đặt mảng 1 chiều

- Cài đặt mảng một chiều:
  - ARRAY : a_1[1, N] OF *datatype* ;
  - **Bảng 2.1:** Kích thước một số kiểu dữ liệu cơ bản trong C/C++

Kiểu DL	Kích thước
char	1
int	2
long, float	4
double	8
char[N]	N

Cài đặt mảng 1 chiều

- ARRAY : a_1[1, N] OF *datatype* ;
- Thứ hai, bố trí các phần tử của mảng vào CTLT đã chọn:
  - Bố trí lần lượt các phần tử của mảng vào trong các ô nhớ của CTLT tuần tự, có nghĩa là phần tử thứ *i* của mảng sẽ được lưu trữ ở ô nhớ thứ *i* ($1 \le i \le N$, với N là kích thước của mảng).
  - Địa chỉ tuyệt đối của phần tử thứ *i,* a1[i]: Ai = A0 + c (i-1)



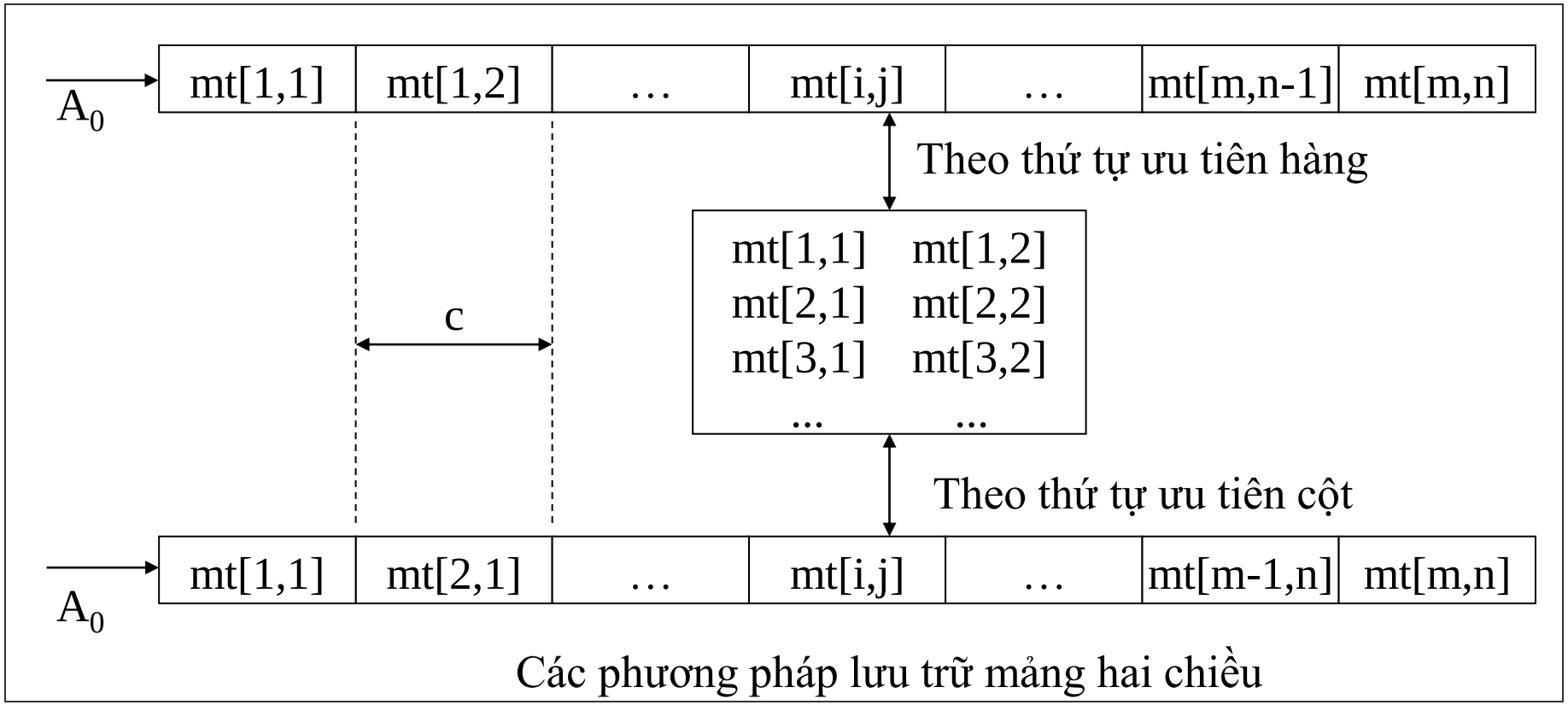
Cài đặt mảng 2 chiều

- Cài đặt mảng hai chiều:
  - ARRAY : a_2[2,M,N] OF *datatype* ;
  - Với các mảng hai hay nhiều chiều hơn, cấu trúc mảng và cấu trúc lưu trữ tuần tự có sự khác biệt đáng kể. Khi CTDL mảng là đa chiều thì cấu trúc lưu trữ tuần tự chỉ có một chiều. Do đó, chúng ta cần phải có một phương pháp để chuyển đổi từ cấu trúc đa chiều sang cấu trúc một chiều và ngược lại.
  - Thứ nhất, xác định các đặc trưng của CTLT tuần tự:
    - Số ô nhớ : bằng M*N, là kích thước của mảng.
    - Kích thước mỗi ô nhớ : là kích thước của *datatype*.
    - Ta cần dành ra một khối nhớ liên tục có kích thước c.M.N, có địa chỉ đầu tiên là A_0 để lưu trữ cho mảng này.

Cài đặt mảng 2 chiều

- Cài đặt mảng hai chiều:
  - ARRAY : a_2[2,M,N] OF *datatype* ;
  - Thứ hai, bố trí các phần tử: có hai phương pháp bố trí:
    - Theo thứ tự ưu tiên hàng: trong phương pháp này, các phần tử của mảng sẽ được bố trí lần lượt theo từng hàng, hết hàng nọ đến hàng kia theo thứ tự các hàng từ trên xuống dưới.
    - Theo thứ tự ưu tiên cột: trong phương pháp này, các phần tử của mảng lại được bố trí theo từng cột, hết cột nọ đến cột kia theo thứ tự các cột từ trái sang phải.

Cài đặt mảng hai chiều: 2 cách bố trí



Các phương pháp lưu trữ mảng hai chiều

Cài đặt mảng 2 chiều

- Cài đặt mảng hai chiều:
  - ARRAY : a_2[2,M,N] OF *datatype* ;
  - *Với phương pháp lưu trữ theo thứ tự ưu tiên hàng*, phần tử a_2[i,j] của mảng được lưu trữ ở ô nhớ thứ z tính theo công thức:
    - z = N*(i-1) + j (2.3)
  - Công thức này được gọi là hàm địa chỉ cho mảng hai chiều a_2 khi các phần tử được bố trí theo thứ tự ưu tiên hàng.
  - Ngược lại, ở ô nhớ z sẽ lưu trữ phần tử a_2[i,j] với i và j được tính theo công thức:
    - i = (z -1) DIV N +1 và j = (z-1) MOD N +1 (2.4)

Cài đặt mảng 2 chiều

- Cài đặt mảng hai chiều:
  - ARRAY : a_2[2,M,N] OF *datatype* ;
  - *Với phương pháp lưu trữ theo thứ tự ưu tiên cột*, phần tử a_2[i,j] của mảng được lưu trữ ở ô nhớ thứ z được tính theo công thức :
    - z = M*(j-1) + i (2.5)
  - Đây là hàm địa chỉ cho mảng hai chiều a_2 khi các phần tử được bố trí theo thứ tự ưu tiên cột.
  - Ngược lại, ở ô nhớ z sẽ lưu trữ phần tử a_2[i,j] với i và j được tính theo công thức:
    - i = (z -1) MOD M +1 và j = (z-1) DIV M +1 (2.6)

Cài đặt mảng nhiều hơn 2 chiều

- Đối với mảng nhiều hơn hai chiều, ta nhớ lại cách chuyển đổi tương đương trong phần trước, là cách chuyển đổi từ các mảng nhiều chiều sang các mảng ít chiều hơn.
- Cụ thể, mảng 3 chiều là một mảng một chiều của các mảng hai chiều. Mà cách lưu trữ mảng 1 chiều và mảng hai chiều ta đã tìm hiểu ở phần trên, nên việc lưu trữ mảng 3 chiều không gặp khó khăn gì.
- Từ cách cài đặt được mảng 3 chiều, chúng ta dễ dang suy ra cách cài đặt các mảng 4, 5,..., N chiều.

Cấu trúc mảng – Hàm địa chỉ

- Khái niệm
  - Hàm địa chỉ có dạng:
    - $f : X \rightarrow Y$ (2.7)
  - Với :
    - X là tập các phần tử của một mảng
    - Y là địa chỉ các ô nhớ của cấu trúc lưu trữ tuần tự
  - Một số quy ước đã sử dụng:
    - c là một số nguyên dương để chỉ kích thước của một ô nhớ
    - A_0 là một số nguyên dương để chỉ địa chỉ đầu tiên của khối lưu trữ, cũng là địa chỉ của ô nhớ đầu tiên.

Cấu trúc mảng – Hàm địa chỉ

- Tính chất
  - Hàm địa chỉ: phụ thuộc vào các đặc trưng của cấu trúc mảng như số chiều, kích thước mảng, kiểu dữ liệu của các phần tử, nhưng đặc trưng số chiều là cơ bản nhất. Chúng ta sẽ thấy rõ hơn về điều này khi xây dựng hàm địa chỉ cho các mảng có số chiều khác nhau.

Cấu trúc mảng – Hàm địa chỉ

- Hàm địa chỉ cho mảng một chiều:
  - ARRAY: vector[1,N] OF *datatype* ;
  - Mảng này được lưu trữ bằng cấu trúc lưu trữ tuần tự.
  - Địa chỉ phần tử thứ *i:* $A_1 = A_0$, $A_2 = A_0 + c$,..., $A_i = A_0 + c(i-1)$;
  - Gọi hàm địa chỉ là f_1 thì nó sẽ có dạng:
    - $f_1(i) = A_0 + c(i-1)$ (2.8)
  - Nhận xét: A_0 và c là các hằng số, hàm chỉ phụ thuộc vào biến số i, là thứ tự của phần tử được lưu trữ - chỉ số của mảng. Vì A_0 là hằng số nên từ nay về sau, để đơn giản các hàm địa chỉ, ta giả sử $A_0 = 0$; Khi đó (2.8) trở thành :
    - $f_1(i) = c(i-1)$ (2.9)

Hàm địa chỉ cho mảng 2 chiều

- Hàm địa chỉ cho mảng hai chiều
  - ARRAY : mt[2,M,N] OF *datatype* ; (2.10)
  - Mảng hai chiều có hai phương pháp lưu trữ là lưu trữ theo thứ tự ưu tiên hàng và theo thứ tự ưu tiên cột. Mỗi phương pháp sẽ có một hàm địa chỉ riêng, nhưng vì cách xây dựng tương tự nhau nên ở đây chúng ta chỉ xét phương pháp lưu trữ theo thứ tự ưu tiên hàng.
  - Theo kết quả (2.3), địa chỉ của phần tử mt[i,j]:
    - $A_{i,j} = c(N(i-1) + j-1)$
  - Hàm địa chỉ của mảng hai chiều được lưu trữ theo thứ tự ưu tiên hàng là:
    - $f_2(i,j) = c(N(i-1) + j-1)$ (2.11)

Hàm địa chỉ cho mảng nhiều hơn 2 chiều

- Hàm địa chỉ cho mảng nhiều hơn hai chiều
  - Theo công thức (2.2)
    - ARRAY: $a_n[N, L_1, L_2, ..., L_n]$ OF datatype ; $\Leftrightarrow$
    - ARRAY: $a_{n-1}[N-1, L_2, ..., L_n]$ OF datatype ; AND
      ARRAY: $a_n[1, L_1]$ OF a_{n-1} ; (2.2)

Hàm địa chỉ cho mảng nhiều hơn 2 chiều

- Để tính hàm địa chỉ của a_n, ta dựa vào hàm địa chỉ của a_{n-1}.
  - Giả sử hàm địa chỉ của a_{n-1} có dạng : $f_{n-1}(i_2,i_3,...,i_n)$ là hàm của n-1 biến, thì hàm địa chỉ của a_n có dạng :

$$f_n(i_1,i_2,...,i_n) = c\prod_{i=2}^{n} L_i(i_1-1) + f_{n-1}(i_2,i_3,...,i_n) \qquad (2.12)$$

$$f_n(i_1,i_2,...,i_n) = c\left[\prod_{i=2}^{n} L_i(i_1-1) + \prod_{i=3}^{n} L_i(i_2-1) + ... + (i_n-1)\right] \qquad (2.13)$$

Hàm địa chỉ cho mảng nhiều hơn 2 chiều

- Hàm địa chỉ cho mảng nhiều hơn hai chiều
  - Chúng ta thử lại với trường hợp mảng hai chiều và ba chiều:
  - Với mảng hai chiều ta có :
    - $f_2(i_1,i_2) = cN(i_1-1) + f_1(i_2)$, với $f_1 = c(i_2-1)$, thay vào ta có:
    - $f_2(i_1,i_2) = cN(i_1-1) + c(i_2-1)$
    - Thay $i=i_1$ và $j=i_2$, ta lại có công thức (2.11)
  - Với mảng ba chiều được khai báo như sau:
    - ARRAY : a_3[3,M,N,P] OF *datatype* ;
    - Ta có công thức hàm địa chỉ như sau:
      $f_3(i_1,i_2,i_3) = c\,[NP(i_1-1) + P(i_2-1) + (i_3-1)]$

2. Cấu trúc danh sách

- Xem phần 2 của chương 2

Thank you!