# Cấu trúc dữ liệu cơ bản

## Danh sách đặc

## Lý thuyết

* Danh sách đặc là mảng 1 chiều cấp phát động.

### Cấu trúc dữ liệu là gì?

### Giải thuật là gì ?

### Danh sách đặc ?

* Là một danh sách mà các phần tử trong danh sách có *cùng kiểu dữ liệu*, được *cấp phát liên tục* trong bộ nhớ.
* Ví dụ khai báo mảng 1 chiều:

Background pattern, icon, rectangle

Description automatically generated with medium confidence

* Max là kích thước thực tế mảng
* n là kích thước hiện tại đang cần dùng
* n có thể thuộc đonạ [1, MAX]
* Danh sách đặc có các thao tác duyệt rất linh hoạt, tuy nhiên kích thước được fix cứng khó thay đổi.

## Một số thao tác với danh sách đặc

### Nhập

* Hàm nhập nếu thực hiện việc nhập số lượng phần tử và các phần tử, thì số lượng thì phải truyền tham chiếu
* Mặc định tham số hình thức mảng là truyền tham chiếu
* Tham số hình thức có thể để trống kích thước

### Xuất

### Tìm

Text

Description automatically generated

* Lưu ý khi viết hàm tìm kiếm, chú ý điều kiện trong *while*
* Điều kiện đầu tiên phải là điều kiện tiên quyết (i < n ): tức là phần tử vẫn nằm trong mảng. Điều kiện thứ 2 là điều kiện quan trọng thứ 2 ( là a[i] khác x). Nếu đảo điều kiện, có thể a[i] sẽ truy xuất ra phần tử ngẫu nhiên bên ngoài mảng, gây ra lỗi không mong muốn.

### Chèn

* Lưu ý: truyền tham chiếu cho kích thước của mảng

### Xóa

### Bài tập vận dụng

Bài 1: Cho mảng 1 chiều, viết chuong trình thực hiện các yêu cầu sau:

a, Nhập, xuất mảng

b, Tìm kiếm 1 phần tử trong mảng. Trả về vị trí đầu tiên xuất hiện hoặc -1 nếu không tồn tại

c, Thêm 1 phần tử

d, Xóa 1 phần tử

## Danh sách liên kết đơn

### Định nghĩa

Danh sách liên kết đơn – linked list là gì ?

Danh sách liên kết đơn là một danh sách mà các phần tử được cấp phát *rời rạc* nhau, và cố định trong bộ nhớ. Mỗi phân tử trong danh sách gồm 2 phần:

Phần 1: vùng thông tin chứa giá trị cần quản lý

Phần 2: vùng liên kết, chứa đại chỉ bộ nhớ của phần tử kế tiếp

|  |  |
| --- | --- |
| Value  9 | Address  3500 |

|  |  |
| --- | --- |
| Value  7 | Address  4000 |

Giả sử danh sách liên kết gồm 9 và 7

9 chứa địa chỉ liên kết 3500 là địa chỉ của 7

7 có địa chỉ 3500

|  |  |
| --- | --- |
| info | link |

Cấu trúc 1 node trong linked list

Background pattern

Description automatically generated with low confidence

Khai báo 1 linked list



Khai báo chỉ cần khai báo phần tử đầu tiên là được.

### Các thao tác cơ bản

#### Khởi tạo

* Khởi tạo danh sách rỗng

Background pattern

Description automatically generated with low confidence

#### Duyệt

* Duyệt danh sách là việc truy xuất giá trị của từng phần tử (bắt đầu từ first)

A picture containing graphical user interface

Description automatically generated

* Gán 1 node tạm vào node đầu tiên, sau đó chạy qua từng danh sách thông qua thông tinh của thánh phần link;

#### Tìm kiếm

* Danh sách liên kết chỉ có con trỏ first được lưu địa chỉ
* Nên chỉ có thể duyệt từ đầu

A picture containing graphical user interface

Description automatically generated

* Duyệt đến cuối danh sách liên kết, p sẽ trỏ vào NULL

#### Thêm

##### Thêm vào đầu

* Giả sử, danh sách đã có (hoặc chưa có phần tử), sẽ có hình dạng như sau

|  |  |
| --- | --- |
|  | ● |

* Việc cần làm là thêm 1 Node mới vào trước danh sách
  + Nếu danh sách là rỗng, thì Node mới sẽ là Node đầu tiên của danh sách luôn
  + Nếu danh sách chưa phải rỗng, thì Node mới cần thêm sẽ liên kết với phần tử đầu tiên của danh sách cũ, và trở thành Node đầu tiên

Diagram

Description automatically generated

Ví dụ: Thêm 1 node có giá trị 30

Bước 1: cấp phát phần tử mới p (có địa chỉ được quản lý bởi biến p)

Text

Description automatically generated with medium confidence

Bước 2: Gán giá trị X (X = 30) cho vùng info của p

Background pattern

Description automatically generated with low confidence

Bước 3: Gán ***giá trị địa chỉ bộ nhớ*** của first cho vùng link của p

(p-> link = first);

Background pattern

Description automatically generated with low confidence

Bước 4: Gán giá trị địa chỉ của p cho biến first

Background pattern

Description automatically generated with low confidence

Hàm chèn vào đầu

Text, letter

Description automatically generated

##### Thêm vào cuối

Diagram

Description automatically generated

###### Trường hợp linked list rỗng (tương đương với thêm vào đầu sanh sách)

###### Trường hợp linked list không rỗng

Bước 1: cấp phát phần tử mới ̣có địac chỉ được quản lý bởi biến p

A picture containing logo

Description automatically generated

Bước 2: Gán giá trị x cho info của p và gán giá trị NULL cho vùng link của p

A picture containing background pattern

Description automatically generated

Bước 3: Tìm phần tử cuối danh sách ̣nếu cosÐ, và gán địa chỉ bộ nhớ của phần tử cuối danh sách cho biến p.

Logo, company name

Description automatically generated

Bước 4: Gán địa chỉ bộ nhớ của p cho vùng link của q

Background pattern

Description automatically generated with low confidence

###### Tổng quát

Text

Description automatically generated

#### Xóa

##### Xóa đầu danh sách

* Xóa hẳn ra memory (không thể tìm lại hay khôi phục)
* Chỉ xóa danh sách khác rỗng
* Ví dụ, muốn xóa node đầu của danh sách sau

A screenshot of a graph

Description automatically generated with low confidence

* Ý tưởng: Nếu xóa phần tử đầu, thì phần tử thứ hai sẽ là phần tử first. Ta gán first cho phần tử thứ 2, đồng thời lưu phần tử thứ nhất(cũ) vào một biến nào đó để không bị mất vết. Tuy nhiên phần tử thứ nhất chưa được xóa, khi đó ta xóa phần tử thứ nhất.

Bước 1: Gán giá trị bộ nhớ của phần tử đầu danh sách cho biến p



Bước 2: Gán địa chỉ bộ nhớ của phần tử thứ 2 trong danh sách cho biến first (first = first->link)

A picture containing logo

Description automatically generated

Bước 3: Thu hồi vùng nhớ phần tử đầu danh sách (delete p)

Background pattern

Description automatically generated with low confidence

Code demo:

Graphical user interface, text

Description automatically generated

##### Xóa cuối danh sách

Diagram

Description automatically generated

* Ý tưởng:
  + Gán liên kết của phần tử kế cuối (15) vào null
  + Xóa phần tử cuối cùng
* Điều kiện lưu ý
  + Danh sách khác rỗng
  + Nếu danh sách chỉ có 1 phần tử, sẽ không tìm được phần tử kế cuối. Trường hợp này, chỉ việc xóa luôn phần tử đó

Bước 1: Tìm địa chỉ bộ nhớ của phần tử cuối danh sách (p) và địa chỉ của phần tử áp cuối danh sách (q) (nếu có).

Text

Description automatically generated

Bước 2: Gán giá trị NULL cho vùng link của q.

A picture containing shape

Description automatically generated

Bước 3: Thu hồi vùng nhớ cuối p;

Background pattern

Description automatically generated with low confidence

* Tuy nhiên, code trên chỉ đúng với trường hợp danh sách có ít nhất 1 phần tử, trong trường hợp tổng quát, có thể code như sau.

Code demo:

A picture containing text

Description automatically generated

##### Xóa sau một phần tử bất kỳ

* Giả sử có danh sách liên kết đơn

Diagram

Description automatically generated

* Yêu cầu, xóa node sau node 15
* Ý tưởng:
  + Tìm node 15
  + Chuyển liên kết từ node 15 đến node sau của node sau node 15 (node 13)
  + Xóa node sau node 15 (node 7)
* Một số lưu ý
  + Danh sách có tồn tại node 15 không ?
  + Node sau node 15 có tồn tại không ? (có thể là node rỗng)

##### Lưu ý

* Phải khởi tạo danh sách trước khi bắt đầu làm việc với danh sách liên kết đơn
* Khi xóa 1 phần tử ra khỏi danh sách, phải dùng toán tử delete để giải phóng vùng nhớ cho hệ thống
* Mọi thao tác trên danh sách liên kết đơn, chỏ cần lưu node đầu tiên first (node này sẽ cho phép ta truy xuất các node tiếp theo).

### Bài tập vận dụng

#### Bài 1: Tạo 1 danh sách liên kết đơn. Thực hiện các thao tác

1. Thêm pt vào đầu
2. Thêm pt vào cuối
3. Tìm kiếm
4. Xuất các phần tử
5. Thêm vào sau 1 phần tử bất kỳ
6. Thêm vào trước 1 phần tử bất kỳ
7. Đếm xem danh sách có b ao nhiêu node
8. Đếm danh sách có bao nhiêu node chứa giá trị số nguyên tố
9. Xóa phần tử đầu sanh sách
10. Xóa phần tử cuối danh sách
11. Xóa phần tử bất kỳ
12. Hủy toàn bộ danh sách sau khi kết thúc chương trình

## Danh sách liên kết kép

### Khái niệm

* Thông thường, người ta ưu tiên dùng danh sách liên kết đơn, vì đỡ phải lưu thêm một vị trí nhớ nữa
* Danh sách kiên kết kép, một node sẽ có 2 liên kết đến node trước và node sau.
* Cấu trúc của 1 node gồm:
  + Info
  + Node \*next
  + Node \*previous

|  |  |  |
| --- | --- | --- |
| Previous | Info | Next |

A picture containing chart

Description automatically generated

### So sánh với các cấu trúc dữ liệu khác

|  |  |  |  |
| --- | --- | --- | --- |
|  | Danh sách liên kết kép | Danh sách liên kết đơn | Danh sách đặc |
| Vị trí các phần tử | Không liền kề nhau | Không liền kề nhau | Đứng cạnh nhau |
| Liên kết các phần tử | Một phần tử gồm 3 thành phần:   * Infor (chứa giá trị) * Con trỏ Next (chứa liên kết đến phần tử phía sau) * Con trỏ Previous(chứa liên kết đến phần tử phía trước) | Một phần tử gồm 2 thành phần:   * Info (chứa giá trị) * Con trỏ chứa liên kết đến phần tử phía sau | Một phần tử chỉ chứa value |
| Các thao tác | * Không linh hoạt bằng danh sách đặc * Linh hoạt hơn danh sách liên kết đơn * Tuy nhiên vẫn ưu tiên dùng danh sách liên kết đơn hơn danh sách liên kết kép (do nó ít hơn 1 bộ nhớ cho con trỏ) |  |  |

### Các thao tác cơ bản

#### Khởi tạo

* Danh sách rỗng là danh sách không có phần tử nào. Do đó, ta gán giá trị NULL cho biến first và last
* Con trỏ first, last phải được khai báo toàn cục trong trương trình

A picture containing text

Description automatically generated

#### Duyệt

* Thuật toán duyệt danh sách liên kết đôi cũng tương tự kỹ thuật duyệt danh sách liên kết đơn. Tuy nhiên danh sách liên kết đôi hỗ trợ con trỏ previous cho phép duyệt từ cuối lên đầu.

##### Duyệt từ đầu

Text

Description automatically generated

##### Duyệt từ cuối

Text

Description automatically generated

#### Tìm kiếm

##### Duyệt từ đầu về cuối

Text, letter

Description automatically generated

##### Duyệt từ cuối về đầu

Text, letter

Description automatically generated

#### Thêm

##### Thêm đầu

* Giả sử, thêm node 30 vào đầu danh sách như hình sau

Chart

Description automatically generated

Bước 1: Cấp phát phần tử mới có địa chỉ được quản lý bởi biến p.

Logo

Description automatically generated with medium confidence

Bước 2: Gán giá trị X (X = 30) cho p->info, và gán con trỏ previous của p về NULL

A picture containing text

Description automatically generated

Bước 3: Gán giá trị địa chỉ bộ nhớ của first cho vùng next của p

Icon

Description automatically generated with low confidence

Bước 4: Hiệu chỉnh giá trị vùng liên kết previous của first thành giá trị của p (first->previous = p)

A picture containing logo

Description automatically generated

Bước 5: Cập nhật lại giá trị của biến first là địa chỉ của phần tử vừa cấp phát p

A picture containing text

Description automatically generated

Code demo:

Text

Description automatically generated

##### Thêm cuối

Chart

Description automatically generated

Bước 1: Cấp phát phần tử mới (có địa chỉ được quản lý bởi biến p)

Logo

Description automatically generated

Bước 2: Gán giá trị x, và gán con trỏ next của p về null

Logo, company name

Description automatically generated with medium confidence

Bước 3: Hiệu chỉnh giá trị vùng liên kết previous của p thành giá trị của last (p->previous = last)



Bước 4: Gán giá trị địa chỉ bộ nhớ của p cho vùng next của last

Text

Description automatically generated with low confidence

Bước 5: Gán giá trị của p cho last

A picture containing background pattern

Description automatically generated

Code demo:

Text

Description automatically generated

#### Xóa

##### Xóa đầu

Chart

Description automatically generated

* Ví dụ, xóa phần tử đầu tiên (10) khỏi danh sách
* Ý tưởng
  + Gán previous của phần tử thứ 2 vào NULL
  + Sau đó delete phần tử Node có info = 10

Chart

Description automatically generated with medium confidence

Bước 1: Khai đáo biến p lưu trữ địa chỉ của phần tử đầu tiên (giá trị biến bằng giá trị biến first).

A picture containing icon

Description automatically generated

Bước 2: Cập nhật giá trị biến first lưu trữ địa chỉ phần tử liền sau phần tử first (first = first->next)



Bước 3: Nếu first khác NULL thì cho previous của first trỏ về NULL (first ->previous = NULL), ngược lại cho last = NULL

Logo, company name

Description automatically generated

Bước 4: Giải phóng vùng nhớ tại p

A picture containing background pattern

Description automatically generated

Code demo:

Text

Description automatically generated

##### Xóa cuối

Chart

Description automatically generated with medium confidence

Chart, scatter chart

Description automatically generated

Ý tưởng:

* Gán 1 biến cho phần tử cuối để lưu vết
* Chuyển liên kết của phần tử kế cuối về NULL
* Xóa phần tử cuối
* Lỗi, khi danh sách không có phần tử nào
* Nếu danh sách có 1 phần tử, sau khi xóa phần tử cuối xong, thì gán first là phần tử kế cuối luông, chứ không thể cập nhật con trỏ previous cho phần tử kế cuối (khi này là phần tử last).

Thuật toán:

Bước 1: Gán giá trị bộ nhớ của phần tử cuối danh sách cho biến p

Chart

Description automatically generated with low confidence

Bước 2: Gán địa chỉ bộ nhớ của phần tử kế cuối trong danh sách cho biến last

Chart

Description automatically generated

Bước 3: Nếu last khác NULL, thì cho vùng next của last trỏ về NULL, ngược lại last = NULL

Logo, company name

Description automatically generated

Bước 4: Giải phóng vùng nhớ tại p

****

Code demo:

Text

Description automatically generated

##### Xóa sau

## Danh sách liên kết vòng