

Chương 3. SẮP XẾP TRÊN DANH SÁCH LIÊN KẾT

ThS. Nguyễn Chí Hiếu

2017

Giới thiệu bài toán sắp xếp trên danh sách liên kết

Phương pháp hoán vị thành phần dữ liệu

Phương pháp thay đổi thành phần liên kết

Giới thiệu bài toán sắp xếp trên danh sách liên kết

Bài toán sắp xếp

- ▶ Sắp xếp là quá trình xử lý các phần tử của một dãy theo đúng thứ tự (thỏa tiêu chuẩn nào đó).
- ▶ Trong bài toán sắp xếp, hai thao tác cơ bản là so sánh và hoán vị giữa hai phần tử trong dãy.

Giới thiệu bài toán sắp xếp trên danh sách liên kết

Bài toán sắp xếp

- ▶ Sắp xếp là quá trình xử lý các phần tử của một dãy theo đúng thứ tự (thỏa tiêu chuẩn nào đó).
- ▶ Trong bài toán sắp xếp, hai thao tác cơ bản là so sánh và hoán vị giữa hai phần tử trong dãy.

Hai phương pháp sắp xếp trên danh sách liên kết

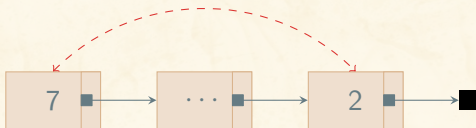
- ▶ Hoán vị *thành phần dữ liệu (Info)* của một nút.
- ▶ Thay đổi *thành phần liên kết (pNext)* của một nút.

Phương pháp hoán vị thành phần dữ liệu

- ▶ Tương tự các thao tác sắp xếp trên mảng.
- ▶ Thao tác hoán vị chính là hoán vị thành phần dữ liệu của mỗi nút.
- ▶ *Thao tác hoán vị đòi hỏi thêm vùng nhớ trung gian chỉ thích hợp với dữ liệu có kích thước nhỏ.*

Phương pháp hoán vị thành phần dữ liệu

- ▶ Tương tự các thao tác sắp xếp trên mảng.
- ▶ Thao tác hoán vị chính là hoán vị thành phần dữ liệu của mỗi nút.
- ▶ *Thao tác hoán vị đòi hỏi thêm vùng nhớ trung gian chỉ thích hợp với dữ liệu có kích thước nhỏ.*



Phương pháp hoán vị thành phần dữ liệu

Nhắc lại thuật toán sắp xếp chọn (*Selection Sort*) thực hiện trên mảng

Thuật toán 1: SelectionSort(a[], n)

- Đầu vào: mảng a gồm n phần tử.
- Đầu ra: mảng a có thứ tự tăng dần.

```
1  for i ← 0 to n - 2
2      min ← i
3      for j ← i + 1 to n - 1
4          if a[j] < a[min]
5              min ← j
6      Swap(a[i], a[min])
```

Phương pháp hoán vị thành phần dữ liệu

Thuật toán sắp xếp chọn (*Selection Sort*) thực hiện trên danh sách liên kết

Thuật toán 2: ListSeletionSort(l)

```
1 // p la nut dau va p khac pTail
2 Khai báo p trở đến pHead
3 while p ≠ pTail
4     Khai báo nút min trở đến nút p
5     // q la nut sau p va q khac rong
6     Khai báo nút q trở đến nút sau p
7     while q ≠ null
8         if Info của nút q < Info của nút min
9             Nút min trở đến nút q
10    Gọi hàm hoán vị Info của hai nút p và min
11    p trở đến p->pNext
```


Phương pháp hoán vị thành phần dữ liệu

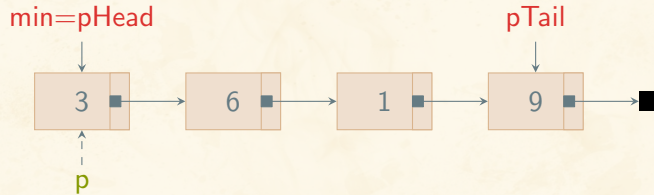
Ví dụ 1

Sắp xếp danh sách liên kết theo thứ tự tăng dần của thành phần dữ liệu.



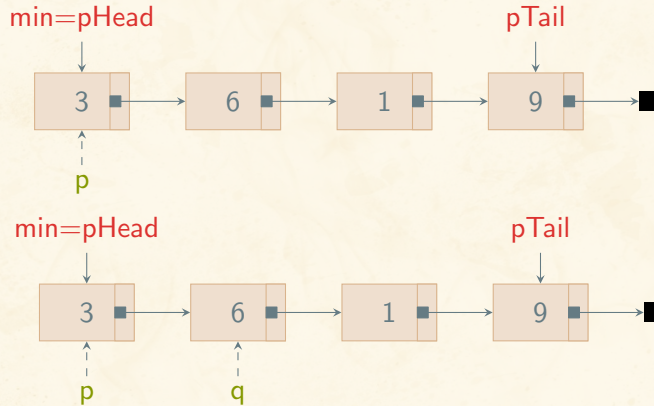
Phương pháp hoán vị thành phần dữ liệu

duyetho



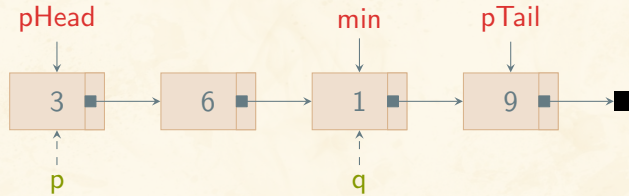
Phương pháp hoán vị thành phần dữ liệu

duy hie



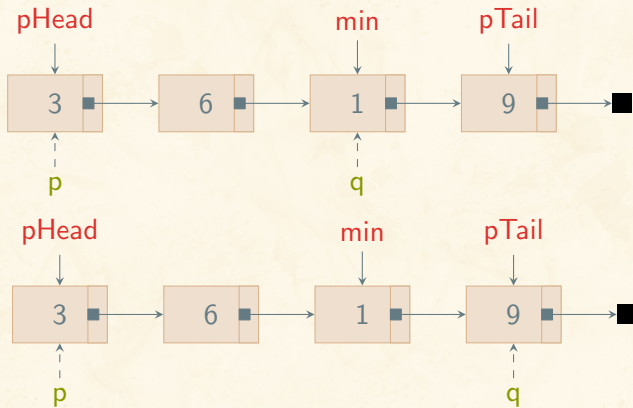
Phương pháp hoán vị thành phần dữ liệu

duy h me



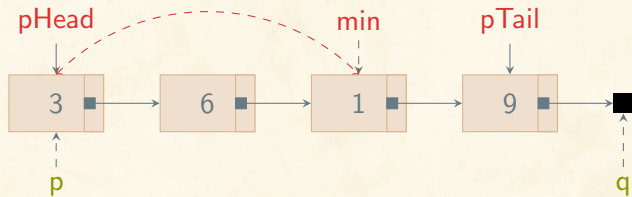
Phương pháp hoán vị thành phần dữ liệu

duyetho



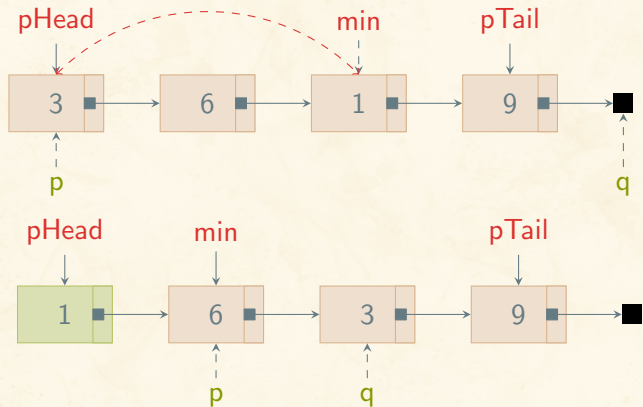
Phương pháp hoán vị thành phần dữ liệu

duy hie



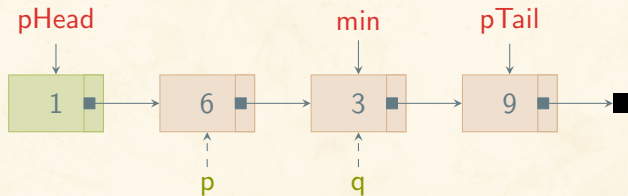
Phương pháp hoán vị thành phần dữ liệu

duyetho



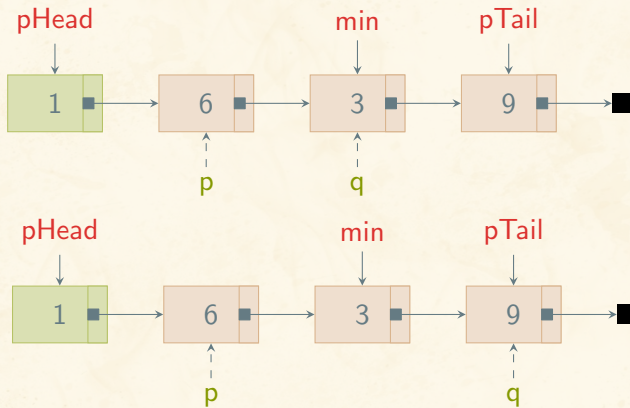
Phương pháp hoán vị thành phần dữ liệu

duy h me



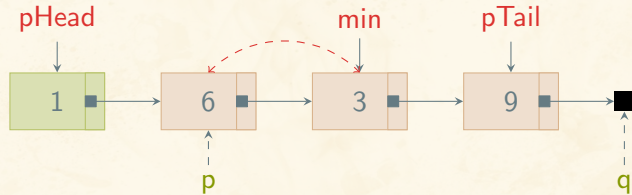
Phương pháp hoán vị thành phần dữ liệu

duyetho



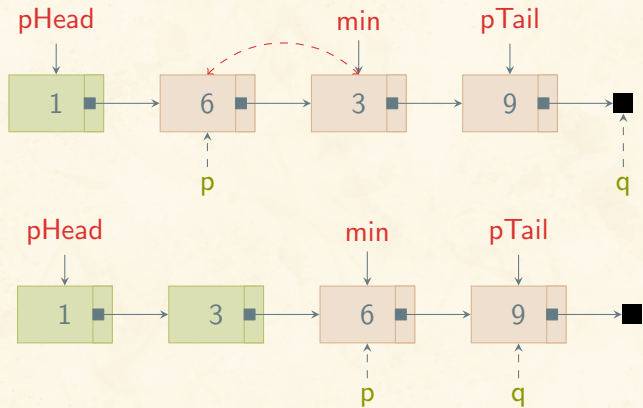
Phương pháp hoán vị thành phần dữ liệu

duyetho



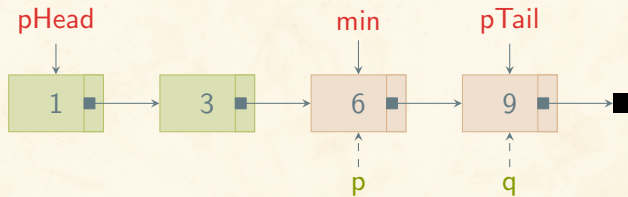
Phương pháp hoán vị thành phần dữ liệu

duyetho



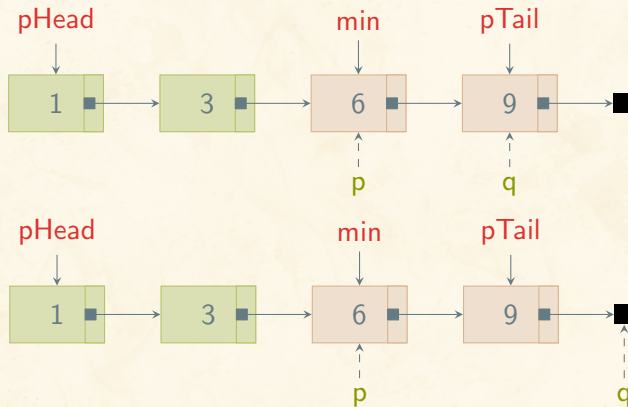
Phương pháp hoán vị thành phần dữ liệu

duyetho

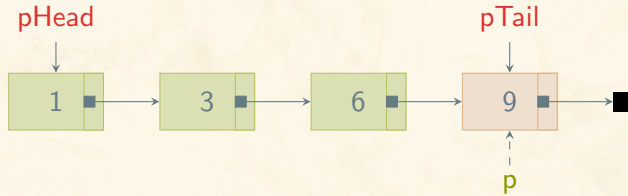


Phương pháp hoán vị thành phần dữ liệu

duyetho

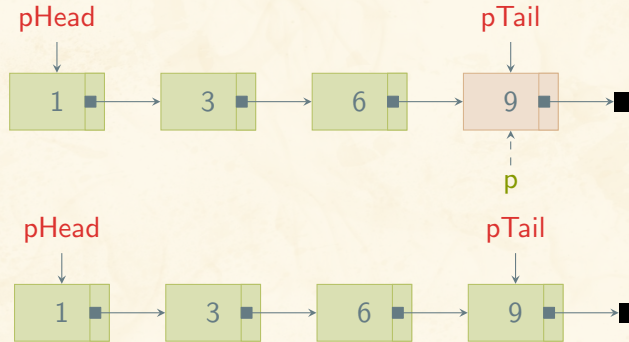


Phương pháp hoán vị thành phần dữ liệu



Phương pháp hoán vị thành phần dữ liệu

duy h me



Phương pháp hoán vị thành phần dữ liệu

```
1 void ListSelectionSort(List *l)
2 {
3     Node *p, *q, *min;
4     p = l->pHead;
5     while (p != l->pTail)
6     {
7         min = p;
8         q = p->pNext;
9         while (q != NULL)
10        {
11            if (q->Info < min->Info)
12                min = q;
13            q = q->pNext;
14        }
15        Swap(p->Info, min->Info);
16        p = p->pNext;
17    }
18 }
```

Phương pháp thay đổi thành phần liên kết

- ▶ Khi sắp thứ tự, chỉ cần thực hiện thao tác *thay đổi thành phần liên kết pNext* giữa các nút.
- ▶ Kích thước của thành phần liên kết không phụ thuộc vào dữ liệu trong danh sách liên kết (4 byte, 8 byte, ... tùy thuộc hệ điều hành).
- ▶ Thuật toán sắp xếp QuickSort, MergeSort thực hiện hiệu quả với danh sách liên kết.

Phương pháp thay đổi thành phần liên kết

Thuật toán sắp xếp nhanh (*QuickSort*)

- ▶ Chọn x là *nút/phần tử đầu* danh sách làm phần tử chốt.
- ▶ Chia danh sách thành hai danh sách con:
 - ▶ Danh sách l_1 : chứa các phần tử *nhỏ hơn hay bằng* x .
 - ▶ Danh sách l_2 : chứa các phần tử *lớn hơn* x .
- ▶ Gọi đệ quy thực hiện với 2 danh sách l_1 và l_2 .
- ▶ Danh sách l sắp theo thứ tự $l_1 \rightarrow x \rightarrow l_2$

Phương pháp thay đổi thành phần liên kết

Thuật toán sắp xếp nhanh (Quick Sort)

Thuật toán 3: ListQuickSort(l)

```
1  if danh sách chỉ chứa 1 nút // Điều kiện dừng đệ quy
2      Dừng thuật toán.
3  x trở đến pHead // x là phần tử pivot
4  pHead trở đến nút sau x // loại x ra khỏi danh sách
5  // 1. Chia: duyệt từ đầu đến cuối danh sách (đã loại x)
6  while chưa duyệt hết danh sách
7      Tách p khỏi danh sách ...
8      if Info của p ≤ Info của x
9          Thêm p vào cuối danh sách l1
10     else
11         Thêm p vào cuối danh sách l2
```

Phương pháp thay đổi thành phần liên kết

duyethoang

```
12  __// 2. Tri: gọi đệ quy hàm  
13  __Gọi đệ quy hàm ListQuickSort() với danh sách con l1  
14  __Gọi đệ quy hàm ListQuickSort() với danh sách con l2  
15  __// 3. Tổng hợp kết quả l: l1->x->l2  
16  __Gọi hàm ListAppend(l, l1, x, l2) nối danh sách l1, l2 và nút x
```


Phương pháp thay đổi thành phần liên kết

Thuật toán 4: ListAppend(l, l1, x, l2)

- Đầu vào: danh sách l, l1, l2 và nút x.
- Đầu ra: danh sách l sau khi nối (l1->x->l2).

```
1  if danh sách l1 rỗng
2    pHead của l trở đến x
3  else
4    Cập nhật pHead của l
5    pTail->pNext của l1 trở đến x
6    x->pNext trở đến pHead của l2
7  if danh sách l2 rỗng
8    pTail của l trở đến x
9  else
10   Cập nhật pTail của l
```

Phương pháp thay đổi thành phần liên kết

duyethoang



Phương pháp thay đổi thành phần liên kết

► $l =$
4, 6, 5, 1, 8, 2

► $x = 4$

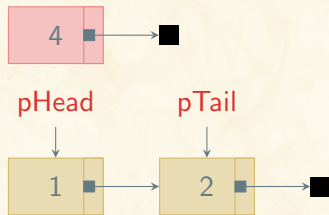
► $l_1 = 1, 2$

► $l_2 = 6, 5, 8$



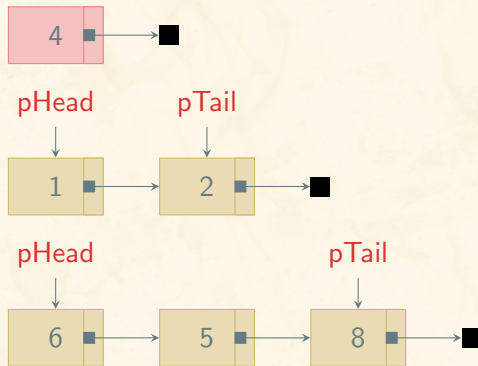
Phương pháp thay đổi thành phần liên kết

- ▶ $l =$
4, 6, 5, 1, 8, 2
- ▶ $x = 4$
- ▶ $l_1 = 1, 2$
- ▶ $l_2 = 6, 5, 8$



Phương pháp thay đổi thành phần liên kết

- ▶ $l =$
4, 6, 5, 1, 8, 2
- ▶ $x = 4$
- ▶ $l_1 = 1, 2$
- ▶ $l_2 = 6, 5, 8$



Phương pháp thay đổi thành phần liên kết

► $l_1 = 1, 2$

► $x = 1$

► $l_{11} = \emptyset$

► $l_{12} = 2$



Phương pháp thay đổi thành phần liên kết

► $l_1 = 1, 2$

► $x = 1$

► $l_{11} = \emptyset$

► $l_{12} = 2$



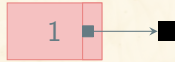
Phương pháp thay đổi thành phần liên kết

► $l_1 = 1, 2$

► $x = 1$

► $l_{11} = \emptyset$

► $l_{12} = 2$



pHead = pTail



Phương pháp thay đổi thành phần liên kết

► $l_1 = l_{11}|x|l_{12}$



pHead = pTail



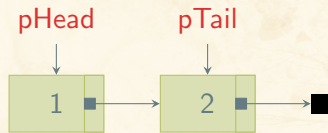
Phương pháp thay đổi thành phần liên kết

duyethoang

► $l_1 = l_{11}|x|l_{12}$



pHead = pTail



Phương pháp thay đổi thành phần liên kết

duy h me

► $l_2 = 6, 5, 8$

► $x = 6$

► $l_{21} = 5$

► $l_{22} = 8$



Phương pháp thay đổi thành phần liên kết

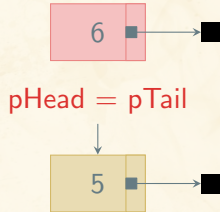
duyethoang

► $l_2 = 6, 5, 8$

► $x = 6$

► $l_{21} = 5$

► $l_{22} = 8$



Phương pháp thay đổi thành phần liên kết

► $l_2 = 6, 5, 8$

► $x = 6$

► $l_{21} = 5$

► $l_{22} = 8$



pHead = pTail



pHead = pTail



Phương pháp thay đổi thành phần liên kết

duyetho

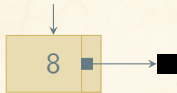
► $l_2 =$
 $l_{21}|x|l_{22}$



pHead = pTail



pHead = pTail



Phương pháp thay đổi thành phần liên kết

► $l_2 =$
 $l_{21}|x|l_{22}$



pHead = pTail

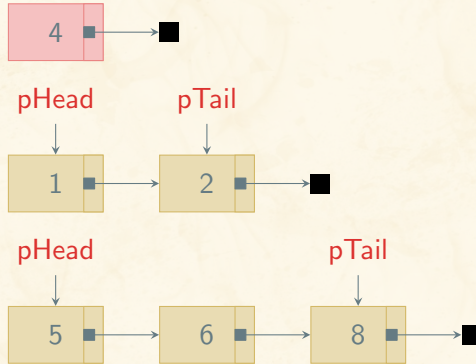


pHead = pTail



Phương pháp thay đổi thành phần liên kết

► $l = l_1 | x | l_2$



Phương pháp thay đổi thành phần liên kết

duyethoang

► $l = l_1 | x | l_2$



Phương pháp thay đổi thành phần liên kết

- ▶ Cài đặt thêm hàm InsertTail(List *l, Node *p) để thêm một nút vào danh sách liên kết.
- ▶ Cài đặt hàm ListQuickSort(List *l).
- ▶ Cài đặt hàm ListAppend(List *l, List *l1, Node *x, List *l2).

```
1 void ListQuickSort(List *l)
2 {
3     __Node *x, *p;
4     __List *l1, *l2;__
5     __if (l->pHead == l->pTail)
6         __return;__
7     __Khởi tạo nút p, danh sách l1, l2 ...__
8     __x = l->pHead;
9     __l->pHead = x->pNext;
```

Phương pháp thay đổi thành phần liên kết

```
10 __while (l->pHead != NULL)
11 __{
12     __p = l->pHead;
13     __l->pHead = p->pNext;
14     __p->pNext = NULL;
15     __if (p->Info <= x->Info)
16         __InsertTail(l1, p);
17     __else
18         __InsertTail(l2, p);
19 __}
20 __ListQuickSort(l1);
21 __ListQuickSort(l2);__
22 __ListAppend(l, l1, x, l2);
23 }
```

Phương pháp thay đổi thành phần liên kết

```
1 void ListAppend(List *l, List *l1, Node *x, List *l2)
2 {
3     if (l1->pHead == NULL)
4         l->pHead = x;
5     else
6     {
7         l->pHead = l1->pHead;
8         l1->pTail->pNext = x;
9     }
10    x->pNext = l2->pHead;
11    if (l2->pHead == NULL)
12        l->pTail = x;
13    else
14        l->pTail = l2->pTail;
15 }
```


Cài đặt các thuật toán sắp xếp trên danh sách liên kết:

1. Thuật toán *SelectionSort*, *InterchangeSort*
2. Thuật toán *QuickSort*.



Dương Anh Đức, Trần Hạnh Nhi.

Nhập môn Cấu trúc dữ liệu và Thuật toán.

Đại học Khoa học tự nhiên TP Hồ Chí Minh, 2003.



Donald E. Knuth.

The Art of Computer Programming, Volume 3.

Addison-Wesley, 1998.



Niklaus Wirth.

Algorithms + Data Structures = Programs.

Prentice-Hall, 1976.



Robert Sedgewick.

Algorithms in C.

Addison-Wesley, 1990.