Mục lục:

ĐỀ CƯƠNG CẤU TRÚC DỮ LIỆU VÀ GIẢI THUẬT

Chương 6: Các giải thuật sắp xếp

Cơ bản	Nâng cao
• Sắp xếp chọn – Selection Sort • Sắp xếp chèn – Insert Sort • Sắp xếp sủi bọt – Bubble Sort	• Sắp xếp nhanh – Quick Sort • Sắp xếp vun đống – Heap Sort • Sắp xếp hòa nhập – Merge Sort

– *Khi tìm hiểu giải thuật thì tìm hiểu 3 nội dung:*

+ Ý tưởng và ví dụ minh họa.

+ Giả mã.

+ Đánh giá giải thuật thông qua độ phức tạp tính toán giải thuật O.

– *Cho bài toán sắp xếp*: Cho dãy khóa là các số nguyên a_1, a_2, …, a_n, được lưu trữ trên mảng 1 chiều. Yêu cầu: Hãy sắp xếp dãy khóa tăng dần theo thứ tự từ trái qua phải.

1. Sắp xếp chọn – Selection Sort

a. Ý tưởng và ví dụ minh họa:

- Ý tưởng:

+ Chọn phần tử có khóa nhỏ nhất.

+ Đổi chỗ nó với phần tử a_1.

+ Sau đó lặp lại thao tác trên với n-1 phần tử còn lại, rồi lặp lại như trên với n-2 phần tử, …, cứ như vậy cho tới khi chỉ còn 1 phần tử.

- Ví dụ minh họa:

• Ví dụ:

Cho dãy khoá ban đầu là: 6, 10, 1, 8, 9

với n=5.

i=1 1, 10, 6, 8, 9

i=2 1, 6, 10, 8, 9

i=3 1, 6, 8, 10, 9

i=4 1, 6, 8, 9, 10

b. Giả mã:

- Vào: Dãy khóa a, số lượng phần tử n.
- Ra: Không có

{Thủ tục này nhận vào dãy khóa a gồm n phần tử được truyền qua đối số. Thực hiện sắp xếp các phần tử của a theo thứ tự tăng dần từ trái qua phải.}

```
Procedure SelectionSort(a, n)
    For i:=1 to n-1 do
        Begin
            {1.Tìm phần tử nhỏ nhất ở vị trí k.}
                +) k:=i;
                +) For j:=i+1 to n do If a[j]<a[k] then k:=j
            {2. Đổi chỗ phần tử nhỏ nhất ở vị trí k cho vị trí i}
                If k≠i then a[k] ↔ a[i]
        End
Return
```

c. Đánh giá giải thuật:

- Giải thuật này có độ phức tạp tính toán là $O(n^2)$.

2. Sắp xếp chèn – Insert Sort

a. Ý tưởng và ví dụ minh họa:

- Ý tưởng:
  + Các phần tử được chia thành dãy đích: a_1, …, a_{i-1} (kết quả). Và dãy nguồn a_i, …, a_n.
  + Bắt đầu với i=2, ở mỗi bước phần tử thứ i ở dãy nguồn được lấy ra và chèn vào vị trí thích hợp trong dãy đích sao cho dãy đích vẫn sắp xếp tăng dần. Sau đó i tăng lên 1 và lặp lại.

- Ví dụ minh họa:

- Ví dụ: Cho dãy khoá 6, 10, 1, 7, 4 với n=5 (dãy số có 5 phần tử).

	Dãy đích	Dãy nguồn
	6	10, 1, 7, 4
i=2	6, 10	1, 7, 4
i=3	1, 6, 10	7, 4
i=4	1, 6, 7, 10	4
i=5	1, 4, 6, 7, 10	

b. Giả mã:

- Vào: Dãy khóa a, số lượng phần tử n.
- Ra: Không có.

{Thủ tục này nhận vào dãy khóa a gồm n phần tử được truyền qua đối số. Thực hiện sắp xếp các phần tử của a theo thứ tự tăng dần từ trái qua phải.}

```
Procedure InsertSort(a, n)
    1)a[0]:-∞;
    2)For i:=2 to n do
        Begin
            +) tg:=a[i]; j:=i-1;
            +) While tg<a[j] do
                Begin
                    a[j+1]:=a[j];
                    j:=j-1;
                End;
            +) a[j+1]:=tg; {Đưa tg vào đúng vị trí, chèn vào sau j}
        End;
```

Return

c. Đánh giá giải thuật:

– Giải thuật này có độ phức tạp tính toán là $O(n^2)$.

3. Sắp xếp nổi bọt – Bubble Sort

a. Ý tưởng và ví dụ minh họa:

- Ý tưởng:
  + So sánh các cặp phần tử liền kề gối nhau từ phải qua trái, nếu phần tử đứng sau nhỏ hơn đứng trước thì đổi chỗ. Kết quả lần thứ nhất, phần tử nhỏ nhất của dãy được đẩy lên vị trí 1 (gọi là phần tử được sắp).
  + Tiếp tục đổi chỗ các phần tử liền kề của dãy chưa sắp, lần thứ 2 ta được phần tử nhỏ nhất của dãy được đưa về vị trí 2.
  + Cứ tiếp tục như vậy làm tương tự như trên cho đến khi dãy chỉ còn 1 phần tử.
- Ví dụ minh họa:

• Ví dụ: Cho dãy khoá ban đầu là: 6, 3, 7, 10, 1, 8 với n=6.

	6, 3, 7, 10, 1, 8
i=1	1, 6, 3, 7, 10, 8
i=2	1, 3, 6, 7, 8, 10
i=3	1, 3, 6, 7, 8, 10
i=4	1, 3, 6, 7, 8, 10
i=5	1, 3, 6, 7, 8, 10

b. Giả mã:

- Vào: Dãy khóa a, số lượng phần tử n.
- Ra: không có.

{Thủ tục này nhận vào dãy khóa a gồm n phần tử được truyền qua đối số. Thực hiện sắp xếp các phần tử của a theo thứ tự tăng dần từ trái qua phải.}

Procedure BubbleSort(a, n)

For i:=1 to n-1 do

For j:=n downto i+1 do

If a[j]<a[j-1] then a[j] ↔ a[j-1];

Return

c. Đánh giá giải thuật:

– Giải thuật này có độ phức tạp tính toán là $O(n^2)$.

4. Sắp xếp nhanh – Quick Sort

a. Ý tưởng và ví dụ minh họa:

- Ý tưởng:
  + Chọn ngẫu nhiên một phần tử x.
  + Duyệt từ bến trái dãy khóa cho tới khi có một phần tử a_i>=x.
  + Sau đó duyệt từ bên phải dãy khóa cho tới khi có một phần tử a_j<=x.
  + Đổi chỗ a_i và a_j.
  + Tiếp tục duyệt và đổi chỗ cho tới khi 2 phía gặp nhau.
  + Kết quả dãy khóa được chia thành 2 phần: Bên trái là các phần tử < x; Bên phải là các phần tử > x.
  + Áp dụng cách tương tự với đoạn bên trái và đoạn bên phải cho tới khi đoạn con chỉ còn 1 phần tử thì dừng.
- Ví dụ minh họa:

b. Giả mã:

- Vào: L, R.
- Ra: không có.

{Thủ tục này nhận vào dãy khóa a gồm n phần tử được truyền qua đối số. Thực hiện sắp xếp các phần tử của a theo thứ tự tăng dần từ trái qua phải.}

Procedure QuickSort(L, R)

1)If L>=R then return;

2)i:=L; j:=R; k:=(L+R) div 2;

3)Repeat

While a[i]<x do i:=i+1;

While a[j]>x do j:=j-1;

If i<j then a[i]↔a[j];

Until i=j

5)Call QuickSort(L, j-1); {Thực hiện trên nửa < x}

6)Call QuickSort(j+1, R); {Thực hiện trên nửa > x}

Return

c. *Đánh giá giải thuật:*

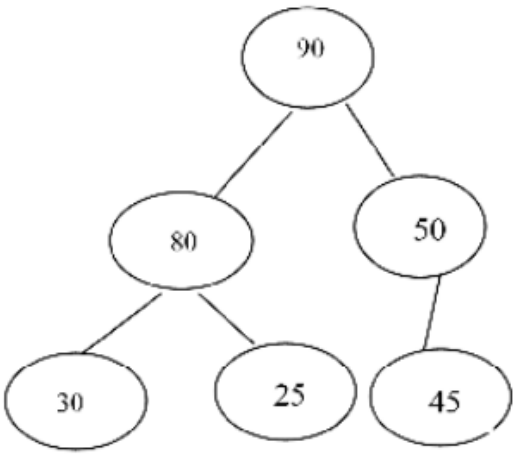
– Giải thuật này có độ phức tạp tính toán là O(n.$\log_2 n$)

5. Sắp xếp vun đống – Heap Sort

a. *Ý tưởng và ví dụ minh họa:*

- Ý tưởng: Được chia thành 2 giai đoạn:
  + *Giai đoạn 1:*
    - Từ dãy khóa ban đầu ánh xạ sang cây nhị phân.
    - Vun cây nhị phân từ dưới lên trên, từ cây con gốc [n/2] về cây gốc 1 để tạo đống.
  + *Giai đoạn 2:*
    - Đổi chỗ nút gốc 1 cho nút n, loại bỏ nút n, hiệu chỉnh lại cây gốc 1 với n-1 nút còn lại.
    - Cứ tiếp tục như vậy cho tới khi cây chỉ còn 1 nút.
- Ví dụ minh họa:

Ví dụ 1: Cây sau đây là một đống.



Khoá ở nút gốc của đống chính là khoá lớn nhất (khoá trội) so với các khoá trên cây.

- Giai đoạn 2:

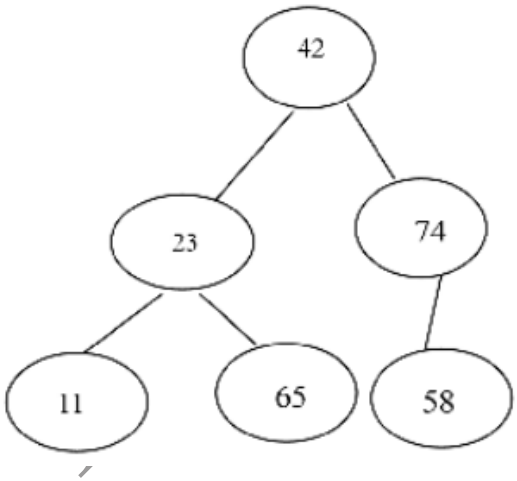
+ Đưa khoá trội về vị trí của nó bằng cách đổi chỗ cho khoá ở vị trí đó.

+ Vun lại đống với cây gồm các khoá còn lại (sau khi đã loại khoá trội)

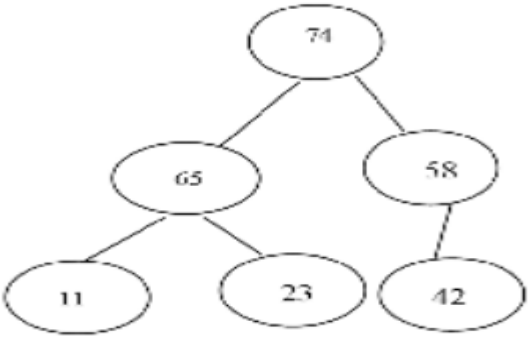
Quá trình trên lại được lặp lại.

Ví dụ: Có dãy khoá 42, 23, 74, 11, 65, 58

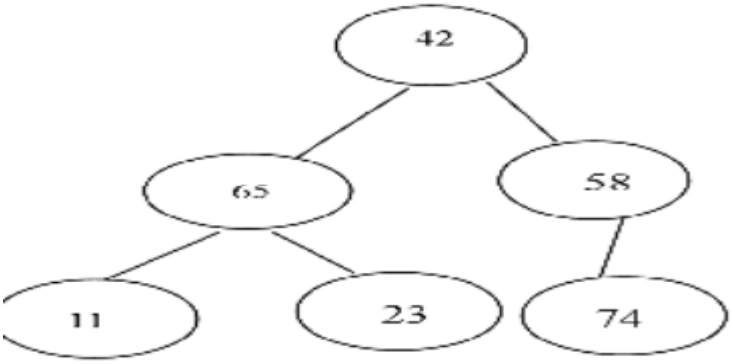
- Ta có cây hoàn chỉnh biểu diễn diễn dãy khoá:



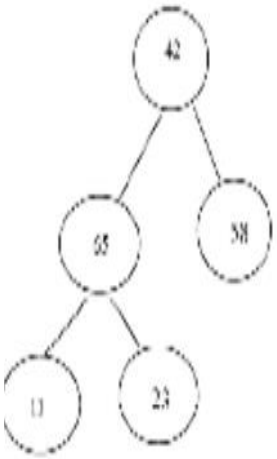
- Giai đoạn đầu: Vun đống ban đầu

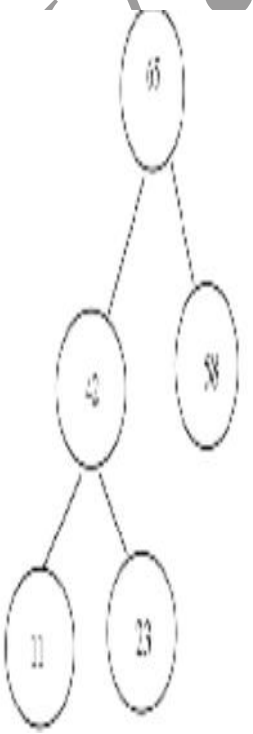


- Giai đoạn 2: Đổi chỗ 74 cho 42

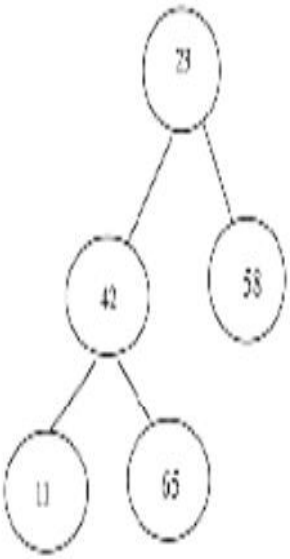


- Loại khoá tội 74 . Dãy đã sắp s= (74)

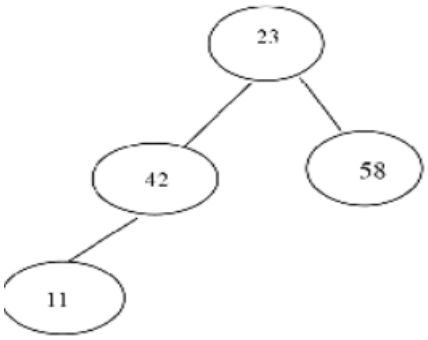




- Đổi chỗ 65 cho 23:



- Loại khoá trội 65 . Dãy đã sắp s= (65, 74)



- Lặp lại các bước tương tự cho các cây còn lại.

Cuối cùng ta thu được dãy đã sắp là s=(11, 23, 42, 58, 65, 74)

b. Giả mã:

- Thủ tục hiệu chỉnh Adjust:
  - + Vào: i, n.
  - + Ra: Không có.

{Thủ tục này hiệu chỉnh cây nhị phân con hoàn chỉnh gốc i trên cây nhị phân có n nút để trở thành "đống" với điều kiện cây con trái và cây con phải có gốc là 2i và 2i+1 đã là đống}

```
Procedure Adjust(i, n)
    1){Khởi đầu}
        Key:=a[i];
        j:=2*i;
    2){Chọn con ứng với khóa lớn nhất trong 2 con của i}
        While j<=n do
            Begin
                +) If (j<n) and (a[j]<a[j+1]) then j:=j+1;
```

3){So sánh khóa cha với khóa lớn nhất}

+) If Key>a[j] then

Begin

a[j/2]:=Key;

Return;

End;

+) a[j/2]:=a[j]; j:=2*i;

End; {Kết thúc While}

4){Đưa Key vào vị trí của nó}

a[j/2]:=Key;

Return

- Thủ tục sắp xếp vun đống Heap Sort:
  + Vào: Dãy khóa a, số lượng phần tử n.
  + Ra: Không có.

{Thủ tục này nhận vào dãy khóa a gồm n phần tử được truyền qua đối số. Thực hiện sắp xếp các phần tử của a theo thứ tự tăng dần từ trái qua phải.}

Procedure HeapSort(a, n)

1.{Tạo đống ban đầu}

For i:=n div 2 downto 1 do Call Adjust(i, n)

2.{Sắp xếp}

For i:=n-1 downto 1 do

Begin

tg:=a[1]; a[1]:=a[i+1]; a[i+1]:=tg;

Call Adjust(1, n);

End;

Return

c. Đánh giá giải thuật:

– Giải thuật này có độ phức tạp tính toán là O(n.$\log_2 n$)

6. Sắp xếp hòa nhập – Merge Sort

6.1. Phép hòa nhập 2 đường: Trộn 2 dãy đã sắp xếp tăng dần thành 1 dãy sắp xếp tăng dần.

a. Ý tưởng và ví dụ minh họa:

- Ý tưởng:

So sánh 2 khóa nhỏ nhất (hoặc lớn nhất của 2 dãy) để đưa vào dãy đích sắp xếp.

Quá trình cứ tiếp tục cho tới khi 1 trong 2 dãy đã cạn. Dãy còn lại đưa nốt sang dãy đích.

- Ví dụ minh họa:

Ví dụ: Dãy 1: (3, 5, 10, 16)
Dãy 2: (1, 4, 15)
Dãy sắp: (1, 3, 4, 5, 10, 15, 16)

b. Giả mã:

Procedure Merge(X, b, m, n, Z)

1)i:=k:=b; j:=m+1;

2)While (i<=m) and (j<=n) do

Begin

+) If x[i]<=a[j] then

Begin

Z[k]:=x[i];

i:=i+1;

End

Else

Begin

Z[k]:=x[j];

j:=j+1;

End

+) k:=k+1;

End

3){Một trong 2 dãy con đã cạn}

If i>m then $(Z_k, \ldots, Z_n):=(x_j, \ldots, x_n)$

Else $(Z_k, \ldots, Z_n):=(x_i, \ldots, x_m)$

Return

6.2. Sắp xếp kiểu hòa nhập trực tiếp:

- Ý tưởng:
  - + Bảng con đã được sắp gọi là một mạch (run).
  - + Mỗi bản ghi coi như 1 mạch có độ dài (kích thước) là 1. Nếu hòa nhập 2 bảng như vậy ta được 1 mạch mới có độ dài = 2. Hòa nhập 2 mạch có độ dài là 2 ta được một mạch có độ dài là 4, …

+ Thủ tục MPass thực hiện một bước của sắp xếp hòa nhập. Nó hòa nhập từng cặp dãy con liền kề nhau, có độ dài L, từ mảng X sang mảng Y, n là số lượng khóa trong X.

- Giả mã thủ tục MPass:

```
Procedure MPass(X, Y, m, L)
    1)i:=1;
    2){Trộn cặp dãy con liền kề có độ dài L}
        While i<=n-(2L-1) do
            Begin
                Call Merge(X, i, i+L-1, i+2L-1, Y);
                i:=i+2L;
            End
    3){Trộn cặp dãy con còn lại cuối cùng với tổng độ dài < 2L}
        If i+L<n then Call Merge(X, i, i+L-1, n, Y)
        Else (y_i, …, y_n):=(x_i, …, x_n)
Return
```

- Giả mã thủ tục sắp xếp kiểu hòa nhập trực tiếp:

```
Procedure MergeSort(X, n)
    1)L:=1;
    2)While L<n do
        Begin
            Call MPass(X, Y, n, L); L:=L+L;
```

Call MPass(Y, X, n, L); L:=L+L;

End

Return

- Ví dụ:

Ví dụ: X=	9, 1, 7, 6, 4, 3, 8, 7		
Bước 1:	1, 9, 6, 7, 3, 4, 7, 8	đưa vào Y	L=1
Bước 2:	1, 6, 7, 9, 3, 4, 7, 8	đưa vào X	L=2
Bước 3:	1, 3, 4, 6, 7, 7, 8, 9	đưa vào Y	L=4

- Đánh giá giải thuật: O($n.\log_2 n$).

Câu hỏi ôn tập cuối chương:

Câu 30: Khi tìm hiểu một giải thuật thì cần tìm hiểu những nội dung nào? Tại sao?

Câu 31: Trình bày 4 giải thuật sắp xếp theo các nội dung của câu 30. Áp dụng chạy giải thuật bằng tay cho dãy sau: …

Câu 32: Hãy trình bày ý tưởng của giải thuật sắp xếp vun đống.