

Chương 1

THUẬT TOÁN

INTERCHANGE SORT

1. BÀI TOÁN DẪN NHẬP

- Bài toán: Viết hàm liệt kê tất cả các cặp giá trị trong mảng một chiều các số nguyên. Lưu ý: cặp (1,2) và cặp (2,1) là giống nhau
- Ví dụ:

12	43	1	34	22
----	----	---	----	----
- Các cặp giá trị trong mảng là:
 - + (12,43),(12,01),(12,34),(12,22)
 - + (43,01), (43,34), (43,22)
 - + (01,34), (01,22)
 - + (34,22)

1. BÀI TOÁN DẪN NHẬP

```
11. void LietKe (int a[], int n)
12. {
13.     for (int i=0; i<=n-2; i++)
14.     {
15.         for (int j=i+1; j<=n-1; j++)
16.         {
17.             printf (" (%d, %d) ",
18.                     a[i], a[j]);
19.         }
20.     }
```

2. NGHỊCH THỂ

- Khái niệm: Một cặp giá trị (a_i, a_j) được gọi là nghịch thể khi a_i và a_j không thỏa điều kiện sắp thứ tự.
- Ví dụ 1: Cho mảng một chiều các số thực a có n phần tử: $a_0, a_1, a_2, \dots, a_{n-2}, a_{n-1}$. Hãy sắp mảng theo thứ tự tăng dần. Khi đó cặp giá trị (a_i, a_j) ($i \leq j$) được gọi là nghịch thể khi $a_i \geq a_j$
- Ví dụ 2: Cho mảng một chiều các số thực a có n phần tử: $a_0, a_1, a_2, \dots, a_{n-2}, a_{n-1}$. Hãy sắp mảng theo thứ tự giảm dần. Khi đó cặp giá trị (a_i, a_j) ($i \leq j$) được gọi là nghịch thể khi $a_i \leq a_j$

2. NGHỊCH THỂ

- Ví dụ 3: Hãy liệt kê các cặp giá trị nghịch thể trong mảng sau, biết rằng yêu cầu là sắp xếp mảng tăng dần

14	29	-1	10	5	23
----	----	----	----	---	----

- Kết quả:
 - + (14, -1), (14,10), (14,5)
 - + (29,-1), (29,10), (29,5), (29,23)
 - + (10,5)

3. TƯ TƯỞNG THUẬT TOÁN

- Thuật toán interchange sort sẽ duyệt qua tất cả các cặp giá trị trong mảng và hoán vị hai giá trị trong một cặp nếu cặp giá trị đó là nghịch thế.

4. ÁP DỤNG THUẬT TOÁN

- Hãy sắp xếp mảng sau tăng dần:

24	45	23	13	43	-1
----	----	----	----	----	----

- Thử tự các bước khi sắp tăng dần mảng trên bằng thuật toán interchange sort.

4. ÁP DỤNG THUẬT TOÁN

– Bước 1:

24	45	23	13	43	-1
24	45	23	13	43	-1
23	45	24	13	43	-1
13	45	24	23	43	-1
13	45	24	23	43	-1
-1	45	24	23	43	13

4. ÁP DỤNG THUẬT TOÁN

– Bước 2:

-1	45	24	23	43	13
-1	24	45	23	43	13
-1	23	45	24	43	13
-1	23	45	24	43	13
-1	13	45	24	43	23

4. ÁP DỤNG THUẬT TOÁN

– Bước 3:

-1	13	45	24	43	23
----	----	----	----	----	----

-1	13	24	45	43	23
----	----	----	----	----	----

-1	13	24	45	43	23
----	----	----	----	----	----

-1	13	23	45	43	24
----	----	----	----	----	----

4. ÁP DỤNG THUẬT TOÁN

– Bước 4:

-1	13	23	45	43	24
----	----	----	----	----	----

-1	13	23	43	45	24
----	----	----	----	----	----

-1	13	23	24	45	43
----	----	----	----	----	----

4. ÁP DỤNG THUẬT TOÁN

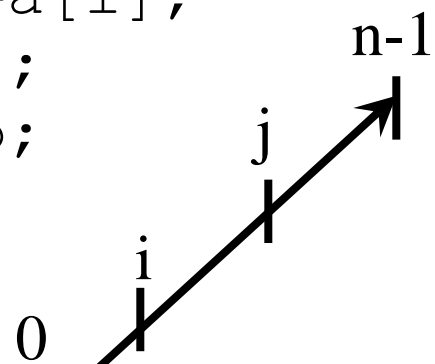
– Bước 5:

-1	13	23	24	45	43
-1	13	23	24	43	45

5. HÀM CÀI ĐẶT

- Bài toán: Định nghĩa hàm sắp mảng một chiều các số nguyên tăng dần bằng thuật toán Interchange sort.
- Hàm cài đặt

```
11. void InterchangeSort (int a[],  
                           int n)  
12. {  
13.     for (int i=0; i<=n-2; i++)  
14.         for (int j=i+1; j<=n-1; j++)  
15.             if (a[i]>a[j])  
16.                 {  
17.                     int temp=a[i];  
18.                     a[i]=a[j];  
19.                     a[j]=temp;  
20.                 }  
21. }
```



5. HÀM CÀI ĐẶT

- Bài toán: Định nghĩa hàm sắp mảng một chiều các phân số tăng dần bằng thuật toán Interchange sort.
- Khai báo kiểu dữ liệu biểu diễn phân số.

```
1. struct phanso
```

```
2. {
```

```
3.     int tu;
```

```
4.     int mau;
```

```
5. };
```

```
6. typedef struct phanso PHANSO;
```

5. HÀM CÀI ĐẶT

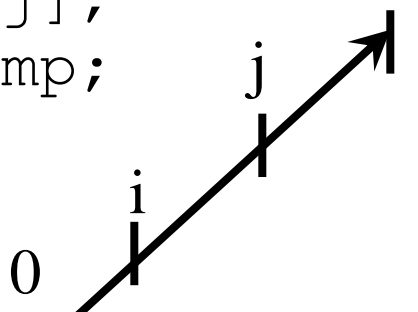
– Định nghĩa hàm

```
1. int SoSanh (PHANSO x,  
              PHANSO y)  
  
2. {  
3.     float a=(float)x.tu/x.mau;  
4.     float b=(float)y.tu/y.mau;  
5.     if (a>b)  
6.         return 1;  
7.     if (a<b)  
8.         return -1;  
9.     return 0;  
10. }
```

5. HÀM CÀI ĐẶT

- Bài toán: Định nghĩa hàm sắp mảng một chiều các phân số tăng dần bằng thuật toán Interchange sort.
- Hàm cài đặt

```
11. void InterchangeSort  
    (PHANSO a[], int n)  
12. {  
13.     for(int i=0; i<=n-2; i++)  
14.         for(int j=i+1; j<=n-1; j++)  
15.             if(SoSanh(a[i], a[j]) == 1)  
16.                 {  
17.                     PHANSO temp = a[i];  
18.                     a[i] = a[j];  
19.                     a[j] = temp;  
20.                 }  
21. }
```



6. DANH SÁCH LIÊN KẾT ĐƠN

- Bài toán: Định nghĩa hàm sắp dslk đơn các số nguyên tăng dần bằng thuật toán Interchange sort.

- Cấu trúc dữ liệu

```
11. struct node
12. {
13. |   int info;
14. |   struct node *pNext;
15. };
16. typedef struct node NODE;
17. struct list
18. {
19. |   NODE *pHead;
20. |   NODE *pTail;
21. };
22. typedef struct list LIST;
```

6.DANH SÁCH LIÊN KẾT ĐƠN

– Hàm cài đặt

```
1. void InterchangeSort (LIST &l)
2. {
3.     for (NODE*p=l.pHead;p->pNext;
           p=p->pNext)
4.         for (NODE*q=p->pNext;q;
               q=q->pNext)
5.             if (p->info>q->info)
6.             {
7.                 int temp=p->info;
8.                 p->info = q->info;
9.                 q->info = temp;
10.            }
11. }
```

7. MỘT CÁCH CÀI ĐẶT KHÁC

- Bài toán: Định nghĩa hàm sắp mảng một chiều các số nguyên tăng dần bằng thuật toán Interchange sort.

- Hàm cài đặt

```
1. void InterchangeSort (int a[],  
                           int n)  
2. {  
3.     for (int i=0; i<=n-2; i++)  
4.         for (int j=n-1; j>=i+1; j--)  
5.             if (a[i]>a[j])  
6.                 {  
7.                     int temp = a[i];  
8.                     a[i] = a[j];  
9.                     a[j] = temp;  
10.                }  
11. }
```

8. DANH SÁCH LIÊN KẾT KÉP

- Bài toán: Định nghĩa hàm sắp dslk kép các số nguyên tăng dần bằng thuật toán Interchange sort.

- Cấu trúc dữ liệu

```
11. struct node
12. {
13.     int info;
14.     struct node *pNext;
15.     struct node *pPrev;
16. };
17. typedef struct node NODE;
18. struct list
19. {
20.     NODE *pHead;
21.     NODE *pTail;
22. };
23. typedef struct list LIST;
```

8. DANH SÁCH LIÊN KẾT KÉP

– Hàm cài đặt 1

```
11. void InterchangeSort (LIST &l)
12. {
13.     for (NODE*p=l.pHead; p->pNext;
           p=p->pNext)
14.         for (NODE*q=p->pNext; q;
               q=q->pNext)
15.             if (p->info>q->info)
16.             {
17.                 int temp = p->info;
18.                 p->info = q->info;
19.                 q->info = temp;
20.             }
21. }
```

8. DANH SÁCH LIÊN KẾT KÉP

– Hàm cài đặt 2

```
11. void InterchangeSort (LIST &l)
12. {
13.     for (NODE*p=l.pHead; p->pNext;
           p=p->pNext)
14.         for (NODE*q= l.pTail; q!=p;
               q=q->pPrev)
15.             if (p->info>q->info)
16.             {
17.                 int temp = p->info;
18.                 p->info = q->info;
19.                 q->info = temp;
20.             }
21. }
```

8. DANH SÁCH LIÊN KẾT KÉP

– Hàm cài đặt 3

```
11. void InterchangeSort (LIST &l)
12. {
13.     for (NODE*p=l.pTail; p->pPrev;
           p=p->pPrev)
14.         for (NODE*q=l.pHead; q!=p;
               q=q->pNext)
15.             if (q->info>p->info)
16.             {
17.                 int temp = p->info;
18.                 p->info = q->info;
19.                 q->info = temp;
20.             }
21. }
```

8. DANH SÁCH LIÊN KẾT KÉP

– Hàm cài đặt 4

```
11. void InterchangeSort (LIST &l)
12. {
13.     for (NODE*p=l.pTail; p->pPrev;
14.          p=p->pPrev)
15.         for (NODE*q=p->pPrev; q;
16.              q=q->pPrev)
17.             if (q->info>p->info)
18.             {
19.                 int temp = p->info;
20.                 p->info = q->info;
21.                 q->info = temp;
22.             }
```


9. PHIÊN BẢN CÀI ĐẶT KHÁC 01

- Bài toán: Định nghĩa hàm sắp mảng một chiều các số nguyên tăng dần bằng thuật toán Interchange sort.
- Hàm cài đặt

```
11. void InterchangeSort (int a[],  
                           int n)  
12. {  
13.     for (int i=n-1; i>=1; i--)  
14.         for (int j=0; j<=i-1; j++)  
15.             if (a[i]<a[j])  
16.                 {  
17.                     int temp=a[i];  
18.                     a[i]=a[j];  
19.                     a[j]=temp;  
20.                 }  
21. }
```

10. PHIÊN BẢN CÀI ĐẶT KHÁC 02

- Bài toán: Định nghĩa hàm sắp mảng một chiều các số nguyên tăng dần bằng thuật toán Interchange sort.
- Hàm cài đặt

```
11. void InterchangeSort (int a[],  
                           int n)  
12. {  
13.     for (int i=n-1; i>=1; i--)  
14.         for (int j=i-1; j>=0; j--)  
15.             if (a[i]<a[j])  
16.                 {  
17.                     int temp=a[i];  
18.                     a[i]=a[j];  
19.                     a[j]=temp;  
20.                 }  
21. }
```

10. PHIÊN BẢN CÀI ĐẶT KHÁC 03

- Bài toán: Định nghĩa hàm sắp mảng một chiều các số nguyên tăng dần bằng thuật toán Interchange sort.
- Hàm cài đặt

```
11. void InterchangeSort (int a[],  
                           int n)  
12. {  
13.     for (int i=0; i<=n-2; i++)  
14.         for (int j=n-1; j>=i+1; j--)  
15.             if (a[i]>a[j])  
16.                 {  
17.                     int temp=a[i];  
18.                     a[i]=a[j];  
19.                     a[j]=temp;  
20.                 }  
21. }
```

