

Chương 1

THUẬT TOÁN

BUBBLE SORT

1. BÀI TOÁN DẪN NHẬP

– Bài toán: Hãy liệt kê các cặp giá trị nằm kế tiếp nhau trong mảng một chiều các số nguyên.

– Ví dụ:

0	1	2	3	4
12	43	1	34	22

– Kết quả: (12,43), (43,1),
(1,34), (34,22)

1. BÀI TOÁN DẪN NHẬP

– Hàm cài đặt

```
1. void LietKe (int a[], int n)
2. {
3.     for (int i=0; i<=n-2; i++)
4.         printf ("%d, %d",
5.                 a[i], a[i+1]);
```

2. TƯ TƯỞNG THUẬT TOÁN

- Bài toán: Cho mảng một chiều a có n phần tử: $a_0, a_1, a_2, \dots, a_{n-2}, a_{n-1}$. Hãy sắp xếp các phần tử trong mảng tăng dần.
- Tư tưởng của thuật toán Bubble Sort là nhẹ nổi lên và nặng chìm xuống.
- Khái niệm nặng nhẹ là khái niệm trừu tượng.

3. ÁP DỤNG THUẬT TOÁN

- Hãy sắp xếp mảng sau tăng dần:

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

- Thứ tự các bước khi sắp tăng dần mảng trên bằng thuật toán bubble sort.

3. ÁP DỤNG THUẬT TOÁN

- Bước 1: Nhẹ nổi lên, nặng chìm xuống lần 1:

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

3. ÁP DỤNG THUẬT TOÁN

- Bước 2: Nhẹ nổi lên, nặng chìm xuống lần 2:

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

3. ÁP DỤNG THUẬT TOÁN

- Bước 3: Nhẹ nổi lên, nặng chìm xuống lần 3:

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

3. ÁP DỤNG THUẬT TOÁN

- Bước 4: Nhẹ nổi lên, nặng chìm xuống lần 4:

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

3. ÁP DỤNG THUẬT TOÁN

- Bước 5: Nhẹ nổi lên, nặng chìm xuống lần 5:

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

4. HÀM CÀI ĐẶT

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

- Hàm cài đặt

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

4. HÀM CÀI ĐẶT

- Bước 1: Nhẹ nổi lên, nặng chìm xuống lần 1:

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

4. HÀM CÀI ĐẶT

- Bước 2: Nhẹ nổi lên, nặng chìm xuống lần 2:

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

4. HÀM CÀI ĐẶT

- Bước 3: Nhẹ nổi lên, nặng chìm xuống lần 3:

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

4. HÀM CÀI ĐẶT

- Bước 4: Nhẹ nổi lên, nặng chìm xuống lần 4:

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

4. HÀM CÀI ĐẶT

- Bước 5: Nhẹ nổi lên, nặng chìm xuống lần 5:

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

4. HÀM CÀI ĐẶT

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

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

4. HÀM CÀI ĐẶT

– Hàm cài đặt cải tiến

```
10. void BubbleSort(int a[], int n)
11. {
12.     for(int i=0; i<=n-2; )
13.     {
14.         int vt = n-1;
15.         for(int j=n-1; j>=i+1; j--)
16.             if(a[j]<a[j-1])
17.             {
18.                 int temp = a[j];
19.                 a[j] = a[j-1];
20.                 a[j-1] = temp;
21.                 vt = j;
22.             }
23.         i = vt;
24.     }
25. }
```

4. HÀM CÀI ĐẶT

- Bài toán: Định nghĩa hàm sắp mảng một chiều các phân số tăng dần bằng thuật toán Bubble 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;
```

4. 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. }
```

4. HÀM CÀI ĐẶT

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

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

5. DANH SÁCH LIÊN KẾT KÉP

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

- Cấu trúc dữ liệu

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

5. DANH SÁCH LIÊN KẾT KÉP

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

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

5. DANH SÁCH LIÊN KẾT KÉP

– Hàm cài đặt

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