



# THUẬT TOÁN BUBBLE SORT

1. Hồ Thái Ngọc
2. ThS. Võ Duy Nguyên
3. TS. Nguyễn Tấn Trần Minh Khang



# BÀI TOÁN DẪN NHẬP 1

# Bài toán dẫn nhập 1



— 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).

# Bài toán dẫn nhập 1



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

— Hàm cài đặt

```
11. void LietKe(int a[], int n)
12. {
13.     for(int i=0; i<=n-2; i++)
14.         cout << "(" << a[i] << "," << a[i+1] << ")";
15. }
```



# BÀI TOÁN DẪN NHẬP 2

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	43	56	41

— Kết quả:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								16	22	66	81	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	43	56	41

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	43	56	41



## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	43	41	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	43	41	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

vt															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	43	41	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt								
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
								22	66	81	99	16	43	41	56	

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

vt															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt								
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
								22	66	81	99	16	41	43	56	

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt								
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
								22	66	81	99	16	41	43	56	



## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

vt															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

vt															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	16	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

vt															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	16	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt								
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
								22	66	81	16	99	41	43	56	

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

vt															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	16	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	16	81	99	41	43	56



## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt								
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
								22	66	16	81	99	41	43	56	

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	16	81	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

vt															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	16	81	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	16	66	81	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

vt															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	16	66	81	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	16	66	81	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	16	66	81	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								16	22	66	81	99	41	43	56



## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								16	22	66	81	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								16	22	66	81	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Ví dụ:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								22	66	81	99	16	43	56	41

— Kết quả:

								vt							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
								16	22	66	81	99	41	43	56

## Bài toán dẫn nhập 2



— Bài toán: Hãy đưa giá trị nhỏ nhất trong đoạn  $[vt..(n - 1)]$  về đầu bằng phương pháp nổi bọt bằng cách duyệt mảng từ cuối mảng về vị trí  $vt$ .

— Hàm cài đặt

```
11. void VeDau(int a[], int n, int vt)
12. {
13.     for(int i=n-1; i>=vt+1; i++)
14.         if(a[i]<a[i-1])
15.             swap(a[i],a[i-1]);
16. }
```



# TƯ TƯỞNG THUẬT TOÁN BUBBLE SORT

# Tư tưởng thuật toán bubble sort



- Tư tưởng của thuật toán Bubble Sort là nhẹ nổi lên và nặng chìm xuống.
- Khái niệm nặng nhẹ là khái niệm trừu tượng.



# THUẬT TOÁN BUBBLE SORT

# Thuật toán bubble sort



- Bước 0: Đưa giá trị nhỏ nhất trong đoạn  $[0..(n - 1)]$  về đầu bằng phương pháp nổi bọt.
- Bước 1: Đưa giá trị nhỏ nhất trong đoạn  $[1..(n - 1)]$  về đầu bằng phương pháp nổi bọt.
- ...
- Bước  $i$ : Đưa giá trị nhỏ nhất trong đoạn  $[i..(n - 1)]$  về đầu bằng phương pháp nổi bọt.
- ...
- Bước  $(n - 2)$ : Đưa giá trị nhỏ nhất trong đoạn  $[(n - 2) ... (n - 1)]$  về đầu bằng phương pháp nổi bọt.





# HÀM CÀI ĐẶT CHUẨN

# Hàm cài đặt chuẩn



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

— Hàm cài đặt

```
11. void HoanVi(int& a, int& b)
12. {
13.     int temp = a;
14.     a = b;
15.     b = temp;
16. }
```

# Hàm cài đặt chuẩn



## — Hàm cài đặt

```
11. void BubbleSort(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[j]<a[j-1])
16.                 swap(a[j], a[j-1]);
17. }
```

Bước  $i$ : Đưa giá trị nhỏ nhất trong đoạn  $[i..(n-1)]$  về đầu bằng phương pháp nổi bọt.



# CHẠY TỪNG BƯỚC THUẬT TOÁN

# Chạy từng bước 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.

# Chạy từng bước thuật toán



— Bước 01: Nhẹ nổi lên, nặng chìm xuống lần 1.

24
45
23
13
43
-1

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

# Chạy từng bước thuật toán



— Bước 01: 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

# Chạy từng bước thuật toán



— Bước 02: 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



# Chạy từng bước thuật toán



— Bước 03: 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

# Chạy từng bước thuật toán



— Bước 04: 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

# Chạy từng bước thuật toán



— Bước 05: Nhẹ nổi lên, nặng chìm xuống lần 5.

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



Bubble sort và mảng một chiều

**PROJECT H01 – DỰ ÁN H01**

# Bubble sort và mảng một chiều



— Viết chương trình thực hiện các yêu cầu sau:

- + Nhập mảng một chiều từ các tập tin: intdata01.inp; intdata02.inp; ...; intdata09.inp; intdata10.inp; intdata11.inp; intdata12.inp; intdata13.inp;
- + Sắp xếp mảng tăng dần bằng thuật toán Bubble sort.
- + Xuất mảng sau khi sắp xếp ra các tập tin: intdata01.out; intdata02.out; ...; intdata09.out; intdata10.out; intdata11.out; intdata12.out; intdata13.out;

# Bubble sort và mảng một chiều



- Định dạng tập tin `intdataxx.inp` và `intdataxx.out`
  - + Dòng đầu tiên: số phần tử của mảng ( $n$ ).
  - + Dòng tiếp theo: lưu  $n$  số nguyên tương ứng với các giá trị trong mảng.

```
*intdata01.inp - Notepad
File Edit Format View Help
10
24 56 53 44 -54 6 63 -47 91 -99
```



# BIẾN THỂ CÀI ĐẶT 01

## Biến thể cài đặt 01



- Tư tưởng của thuật toán Bubble Sort là nhẹ nổi lên và nặng chìm xuống.
- Tư tưởng của thuật toán Bubble Sort biến thể 1 là nặng chìm xuống và nhẹ nổi lên.
- Khái niệm nặng nhẹ là khái niệm trừu tượng.



# Biến thể cài đặt 01



— Hàm cài đặt

```
11. void BubbleSort01(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[j]>a[j+1])
16.                 swap(a[j], a[j+1]);
17. }
```



Bubble sort và mảng một chiều

**PROJECT H02 – DỰ ÁN H02**

# Bubble sort và mảng một chiều



— Viết chương trình thực hiện các yêu cầu sau:

- + Nhập mảng một chiều từ các tập tin: intdata01.inp; intdata02.inp; ...; intdata09.inp; intdata10.inp; intdata11.inp; intdata12.inp; intdata13.inp;
- + Sắp xếp mảng tăng dần bằng thuật toán Bubble sort với biến thể 1.
- + Xuất mảng sau khi sắp xếp ra các tập tin: intdata01.out; intdata02.out; ...; intdata09.out; intdata10.out; intdata11.out; intdata12.out; intdata13.out;

# Bubble sort và mảng một chiều



- Định dạng tập tin `intdataxx.inp` và `intdataxx.out`
  - + Dòng đầu tiên: số phần tử của mảng ( $n$ ).
  - + Dòng tiếp theo: lưu  $n$  số nguyên tương ứng với các giá trị trong mảng.

```
*intdata01.inp - Notepad
File Edit Format View Help
10
24 56 53 44 -54 6 63 -47 91 -99
```



# Thuật toán Bubble sort và mảng cấu trúc

## **MẢNG CẤU TRÚC**

# Bubble sort và mảng cấu trúc



- 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.

```
11.struct phanso
12.{
13.|    int tu;
14.|    int mau;
15.};
16.typedef struct phanso PHANSO;
```

# Bubble sort và mảng cấu trúc



```
11. int SoSanh(PHANSO x, PHANSO y)
12. {
13.     float a = (float)x.tu/x.mau;
14.     float b = (float)y.tu/y.mau;
15.     if(a>b)
16.         return 1;
17.     if(a<b)
18.         return -1;
19.     return 0;
20. }
```

# Bubble sort và mảng cấu trúc



```
11. void BubbleSort(PHANSO 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(SoSanh(a[j], a[j-1]) == -1)
16.             {
17.                 PHANSO temp = a[j];
18.                 a[j] = a[i];
19.                 a[i] = temp;
20.             }
21. }
```





Bubble sort và mảng cấu trúc

## **PROJECT H03 – DỰ ÁN H03**

# Bubble sort và mảng cấu trúc



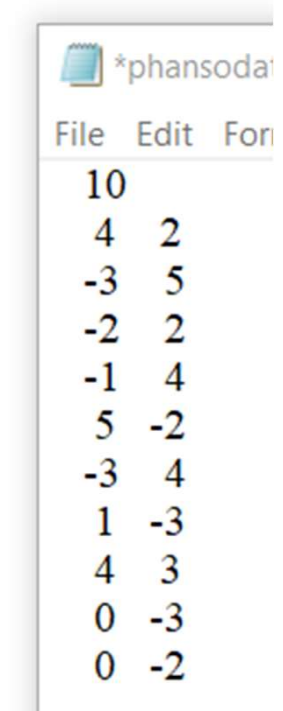
— Viết chương trình thực hiện các yêu cầu sau:

- + Nhập mảng một chiều từ các tập tin: phansodata01.inp; phansodata02.inp; ...; phansodata09.inp; phansodata10.inp; phansodata11.inp; phansodata12.inp; phansodata13.inp;
- + Sắp xếp mảng phân số tăng dần bằng thuật toán Bubble sort.
- + Xuất mảng sau khi sắp tăng ra các tập tin: phansodata01.out; phansodata02.out; ...; phansodata09.out; phansodata10.out; phansodata11.out; phansodata12.out; phansodata13.out;

# Bubble sort và mảng cấu trúc



- Định dạng tập tin
  - + phansodatxx.inp,
  - + phansodatxx.out
- Dòng đầu tiên: số phần tử của mảng ( $n$ ).
- $n$  dòng tiếp theo: mỗi dòng lưu hai số nguyên tương ứng với phân số trong mảng.



10	
4	2
-3	5
-2	2
-1	4
5	-2
-3	4
1	-3
4	3
0	-3
0	-2



# Thuật toán Bubble sort và ma trận

## **MA TRẬN**

# Bubble sort và ma trận



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

	0	1	2	3
0	89	12	78	91
1	61	37	8	18
2	78	23	35	22

Ma trận trước khi sắp tăng

	0	1	2	3
0	8	12	18	22
1	23	35	37	61
2	78	78	89	91

Ma trận sau khi sắp tăng

# Bubble sort và ma trận



— Định nghĩa hàm

```
11. void BubbleSort(int a[][100], int m, int n)
12. {
13.     for(int i=0; i<=m*n-2; i++)
14.         for(int j=m*n-1; j>=i+1; j--)
15.             if(a[(j-1)/n][(j-1)%n] > a[j/n][j%n])
16.                 swap(a[j/n][j%n], a[(j-1)/n][(j-1)%n]);
17. }
```



Bubble sort và ma trận

# **PROJECT H04 – DỰ ÁN H04**

# Bubble sort và ma trận



— Viết chương trình thực hiện các yêu cầu sau:

- + Nhập ma trận các số nguyên từ các tập tin: intmatran01.inp; intmatran02.inp; ...; intmatran09.inp; intmatran10.inp; intmatran11.inp; intmatran12.inp; intmatran13.inp;
- + Sắp xếp ma trận tăng dần bằng thuật toán Bubble sort.
- + Xuất ma trận sau khi sắp xếp ra các tập tin: intmatran01.out; intmatran02.out; ...; intmatran09.out; intmatran10.out; intmatran11.out; intmatran12.out; intmatran13.out;



# Bubble sort và ma trận



- Định dạng tập tin
  - + `intmatranxx.inp` và
  - + `intmatranxx.out`
- Dòng đầu tiên: lưu hai số nguyên tương ứng với số hàng ma trận ( $m$ ) và số cột ma trận ( $n$ ).
- $m$  dòng tiếp theo: mỗi dòng lưu  $n$  số nguyên tương ứng với các giá trị trong ma trận.

\*intmatrandata01.inp - Notepad

File Edit Format View Help

```
8 10
-27 63 -7 -49 78 -27 92 -71 71 59
14 20 32 -81 -17 -12 49 -38 17 78
-71 77 33 -62 49 39 47 -23 -2 -20
89 -89 -39 0 3 97 67 -98 -32 -33
67 82 20 -9 4 -3 -1 -87 -54 61
-68 -6 -86 52 9 -80 84 -99 5 18
-54 -19 18 100 -73 -40 -58 -80 -12 -96
-43 44 32 -85 -11 32 77 33 -75 -32
```

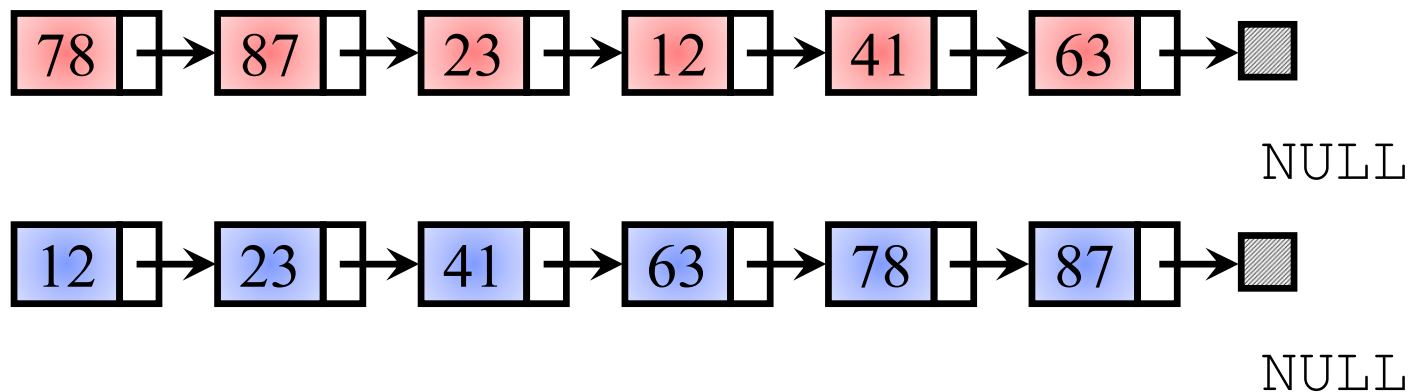


Thuật toán Bubble sort và dslk đơn  
**DANH SÁCH LIÊN KẾT ĐƠN**

# Bubble sort và dslk đơn



- Bài toán: Định nghĩa hàm sắp xếp danh sách liên kết đơn các số nguyên tăng dần bằng thuật toán Bubble sort.





Thuật toán Bubble sort và dslk đơn

**PROJECT H05 – DỰ ÁN H05**

# Thuật toán Bubble sort và dslk đơn



— Viết chương trình thực hiện các yêu cầu sau:

- + Nhập dslk đơn các số nguyên từ các tập tin: intdata01.inp; intdata02.inp; ...; intdata09.inp; intdata10.inp; intdata11.inp; intdata12.inp; intdata13.inp;
- + Sắp xếp dslk đơn các số nguyên tăng dần bằng thuật toán Bubble sort.
- + Xuất dslk đơn các số nguyên sau khi sắp xếp ra các tập tin: intdata01.out; intdata02.out; ...; intdata09.out; intdata10.out; intdata11.out; intdata12.out; intdata13.out;

# Thuật toán Bubble sort và dslk đơn



- Định dạng tập tin `intdataxx.inp` và `intdataxx.out`
  - + Dòng đầu tiên: số phần tử của dslk đơn các số nguyên ( $n$ ).
  - + Dòng tiếp theo: lưu  $n$  số nguyên tương ứng với các giá trị trong dslk đơn các số nguyên.

```
*intdata01.inp - Notepad
File Edit Format View Help
10
24 56 53 44 -54 6 63 -47 91 -99
```

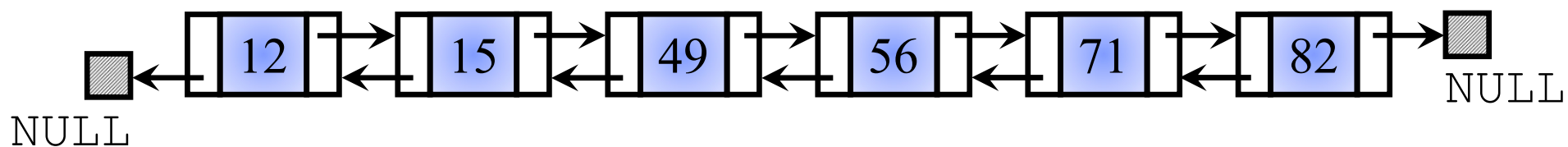
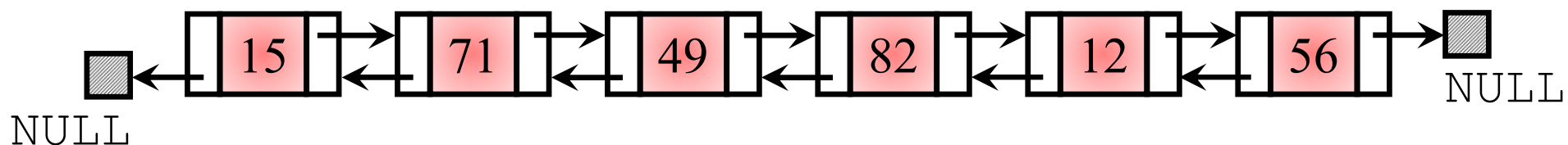


Thuật toán Bubble sort và dslk kép  
**DANH SÁCH LIÊN KẾT KÉP**

# Bubble sort và dslk kép



- Bài toán: Định nghĩa hàm sắp xếp danh sách liên kết kép các số nguyên tăng dần bằng thuật toán Bubble sort.







# PROJECT H06 – DỰ ÁN H06

# Bubble sort và dslk kép



— Viết chương trình thực hiện các yêu cầu sau:

- + Nhập dslk kép các số nguyên từ các tập tin: intdata01.inp; intdata02.inp; ...; intdata09.inp; intdata10.inp; intdata11.inp; intdata12.inp; intdata13.inp;
- + Sắp xếp dslk kép các số nguyên tăng dần bằng thuật toán Bubble sort.
- + Xuất dslk kép các số nguyên sau khi sắp xếp ra các tập tin: intdata01.out; intdata02.out; ...; intdata09.out; intdata10.out; intdata11.out; intdata12.out; intdata13.out;

# Bubble sort và dslk kép



- Định dạng tập tin `intdataxx.inp` và `intdataxx.out`
  - + Dòng đầu tiên: số phần tử của dslk kép các số nguyên ( $n$ ).
  - + Dòng tiếp theo: lưu  $n$  số nguyên tương ứng với các giá trị trong dslk kép các số nguyên.

```
*intdata01.inp - Notepad
File Edit Format View Help
10
24 56 53 44 -54 6 63 -47 91 -99
```



# ĐỘ PHỨC TẠP CỦA THUẬT TOÁN

# Độ phức tạp của thuật toán



— Hãy đánh giá độ phức tạp của thuật toán Bubble sort dựa trên hàm cài đặt chuẩn.

```
11. void BubbleSort(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[j]<a[j-1])
16.                 swap(a[j], a[j-1]);
17. }
```

# Độ phức tạp của thuật toán





Thuật toán Bubble sort

**ĐẶC ĐIỂM – ĐIỂM MẠNH – ĐIỂM YẾU**

# Đặc điểm – điểm mạnh – điểm yếu



— Đặc điểm thuật toán bubble sort:

- + Độ phức tạp về thời gian (time complexity):  $O(n^2)$ .
- + Độ phức tạp về bộ nhớ (space complexity):  $O(1)$ .
- + Trường hợp xấu nhất (worst case):  $O(n^2)$ .
- + Trường hợp trung bình (average case):  $O(n^2)$ .
- + Trường hợp tốt nhất (best case):  $O(n)$ .
- + Ổn định.



# Đặc điểm – điểm mạnh – điểm yếu



## — Điểm mạnh:

- + Thuật toán rõ ràng, dễ hiểu.
- + Thuật toán dễ cài đặt.
- + Không yêu cầu dung lượng bộ nhớ lớn.



# Đặc điểm – điểm mạnh – điểm yếu



## — Điểm yếu:

- + Thời gian thực hiện thuật toán lâu.
- + Không nhận biết mảng đã được sắp xếp.



**Cảm ơn quý vị đã lắng nghe**

**Nhóm tác giả**

**Hồ Thái Ngọc**

**ThS. Võ Duy Nguyên**

**TS. Nguyễn Tấn Trần Minh Khang**