

Mảng - Array

Mảng – Array

- Một số tính chất
- Khai báo mảng trong C
- Truy xuất các thành phần
- Truyền tham số kiểu mảng cho hàm
- Một số thao tác cơ sở
- Mảng nhiều chiều

Mảng – Một số tính chất

- Mảng là một kiểu dữ liệu có cấu trúc do người lập trình định nghĩa
- Dùng biểu diễn các đối tượng dữ liệu ở dạng một dãy các thành phần có cùng kiểu với nhau – kiểu cơ sở
- NNLT C luôn chỉ định một khối nhớ liên tục cho một biến kiểu mảng
- Kích thước của mảng được xác định ngay khi khai báo và không bao giờ thay đổi

Mảng – Khai báo trong C

kiểu **có** **sở** **TênBiên** [**Số** **thành** **phần**] ;

kiểu của mỗi thành phần

do lập trình viên đặt tên

hàng số, số thành phần
tối đa của mảng

int

a [**100**] ;

//a là mảng biểu diễn dãy gồm 100 thành phần int

Mảng – Ví dụ

```
#define SIZE      10  
  
int      a[5];      // a dãy gồm 5 số nguyên  
long int big[100]; // big: chiếm 400 bytes!  
double   d[100];   // d: chiếm 800 bytes!  
long double v[SIZE];// v:10 long doubles
```

Mảng – Ví dụ

```
int      a[5]      = { 10, 20, 30, 40, 50};  
double  d[100]     = { 1.5, 2.7};  
short   primes[]  = { 1, 2, 3, 5, 7, 11, 13};  
long    b[50]      = { 0 };
```

Trình biên dịch xác
định kích thước
gồm 7 thành phần

khởi tạo cho
5 thành phần

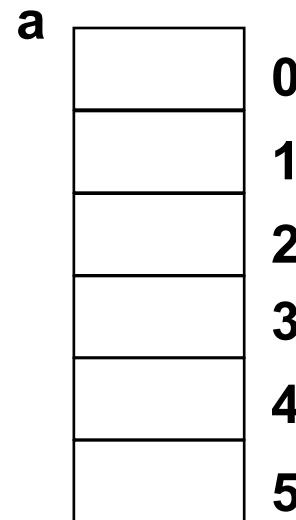
2 thành phần
đầu tiên được
khởi tạo, phần
còn lại: 0

cách nhanh nhất để
khởi tạo tất cả các
thành phần bằng 0

Mảng – Truy xuất các phần tử

- Các thành phần của mảng được truy xuất thông qua chỉ số của chúng 0..n-1
- Thao tác truy xuất không kiểm tra giới hạn của chỉ số

```
int main()
{
    int a[6];
    int i = 7;
    a[0] = 59;
    a[5] = -10;
    a[i/2] = 2;
    a[6] = 0;    ←
    a[-1] = 5;   ←
    return 0;
}
```



Truyền tham số Mảng cho hàm

- Tham số kiểu mảng được truyền cho hàm chính là địa chỉ của phần tử đầu tiên trên mảng
- Số thành phần trong tham số mảng có thể để trống.
- Số thành phần thực sự được sử dụng phải truyền qua một tham số khác (vd: size)



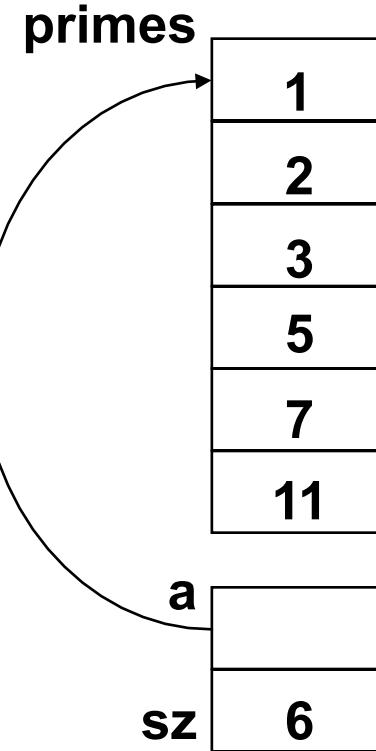
```
int add_elements(int a[], int size)
{
```

```
int add_elements(int *p, int size)
{
```

Ví dụ

```
#include <stdio.h>
void sum(long [], int);
int main(void) {
    long primes[6] = { 1, 2,
                      3, 5, 7, 11 };
    sum(primes, 6);
    printf("%li\n", primes[0]);
    return 0;
}

void sum(long a[], int sz) {
    int i;
    long total = 0;
    for(i = 0; i < sz; i++)
        total += a[i];
    a[0] = total;
}
```



dùng để kiểm tra
giới hạn chỉ số

tổng được lưu vào
phần tử đầu tiên

Một số thao tác cơ sở

- Nhập
- Xuất
- Thêm một thành phần dữ liệu
- Loại bỏ một thành phần dữ liệu
- Tìm kiếm
- Sắp xếp

Mảng – Nhập dữ liệu

```
void ReadData(int a[], int size)
{
    int i;
    for(i = 0; i < size; i++)
    {
        printf("Nhập thành phần %d: ", i);
        scanf("%d", &a[i]);
    }
}
```

duyệt qua tất cả các phần tử

nhập dữ liệu cho a[i]

Mảng – Xuất dữ liệu ra màn hình

```
void WriteData(int a[], int size)
{
    int i;

    for(i = 0; i < size; i++)
        printf("%d ", a[i]);
    printf("\n");
}
```

Mảng – Nhập xuất dữ liệu

```
#include <stdio.h>
void ReadData(int [], int );
void WriteData(int [], int );
int main()
{
    int a[100], n;

    printf("Nhập số thành phần của dãy: ");
    scanf("%d", &n);
    printf("Nhập các thành phần của dãy: ");
    ReadData(a, n);
    printf("Dãy vừa nhập: \n");
    WriteData(a, n);
    return 0;
}
```

Mảng – Tìm vị trí X trong dãy

- Bài toán: Tìm vị trí X trên mảng a đang có N thành phần.
- Giải pháp: Tìm tuần tự

```
//input: dãy (a, N), X
//output: Vị trí của X, -1 nếu không có

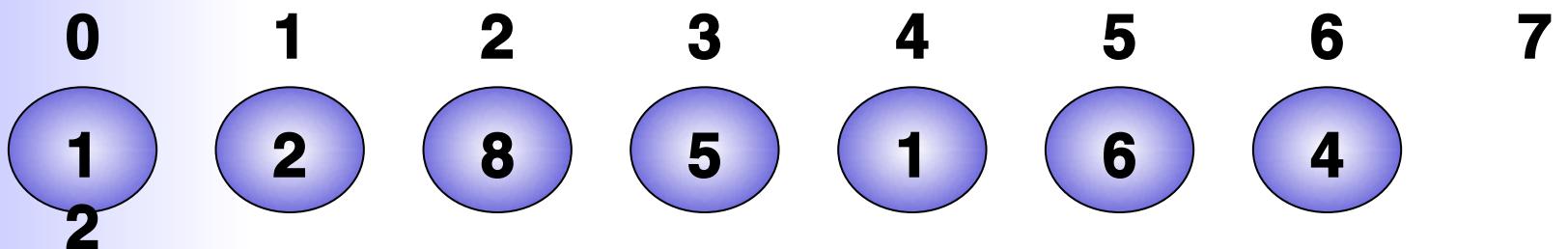
int      Search(int a[], int N, int X)
{
    for (int i = 0; i < N; i++)
        if (a[i] == X)
            return i;
    return -1;
}
```

Mảng – Thêm một thành phần dữ liệu

- Bài toán: cần thêm thành phần dữ liệu X vào mảng a đang có N thành phần.
- Hai trường hợp cần xem xét:
 - Dãy chưa có thứ tự
 - Thêm X vào cuối a.
 - Dãy đã có thứ tự
 - Tìm vị trí thích hợp, chèn X vào

Mảng – Thêm X vào cuối dãy

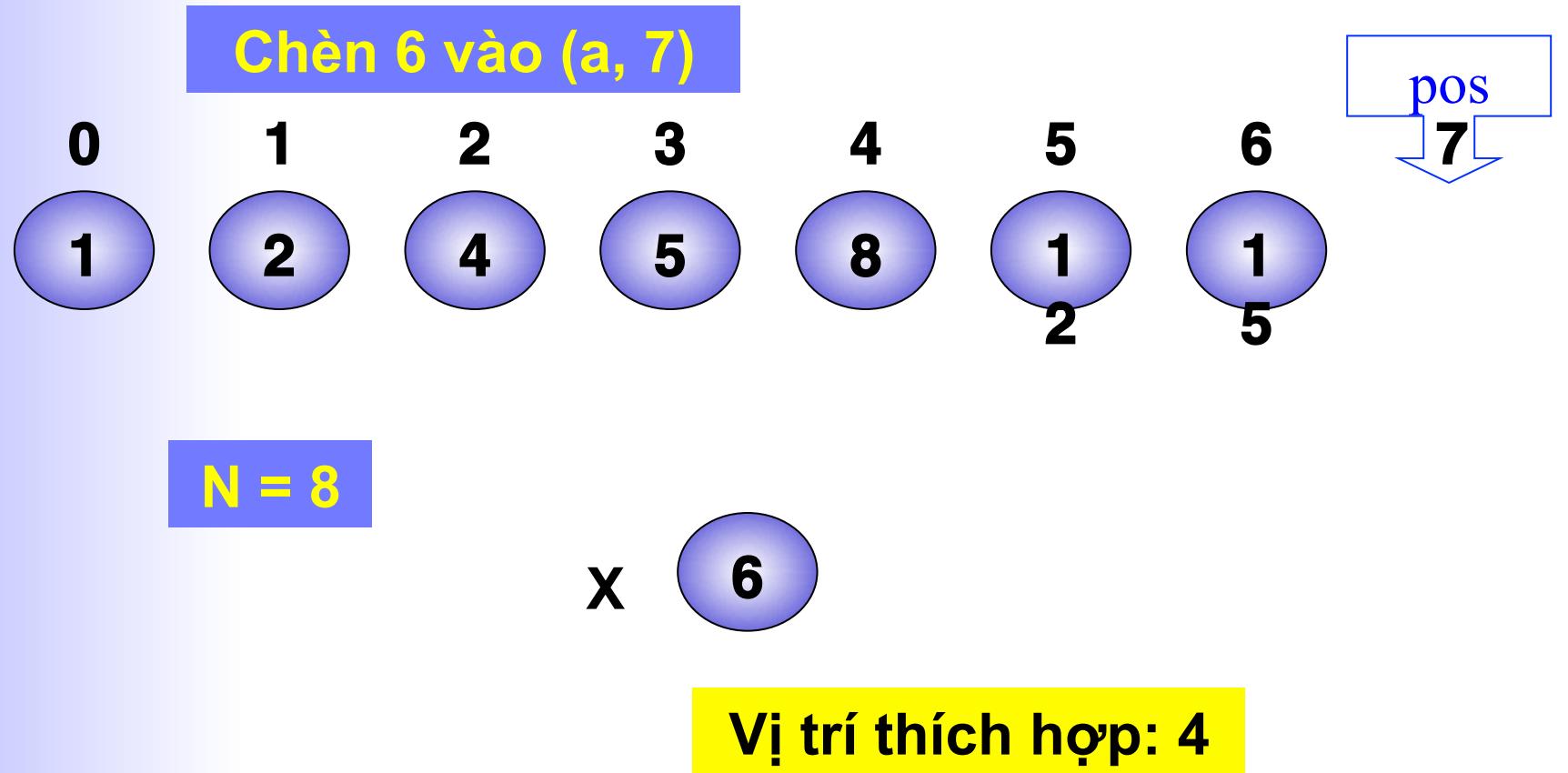
Thêm 15 vào (a, 7)



N = 8

```
a[N] = x;  
N++;
```

Mảng – Chèn X vào dây tăng dần



Mảng – Chèn X vào dãy tăng dần

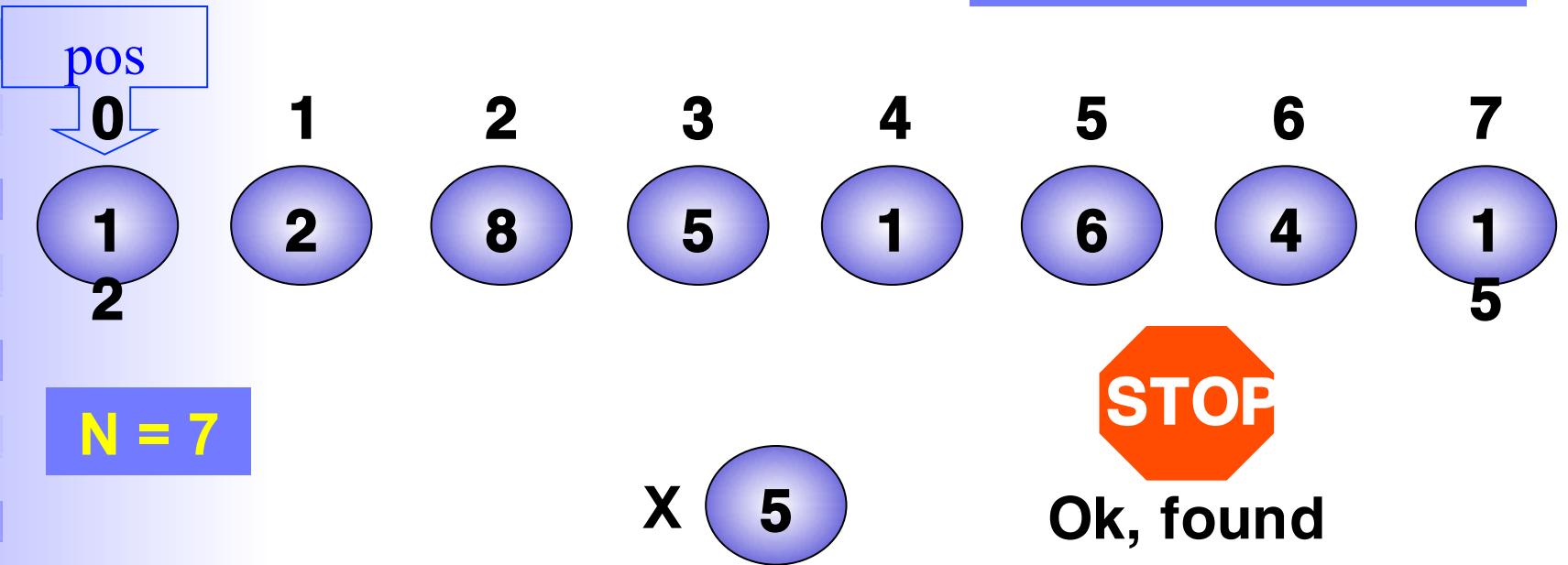
```
//input: dãy (a, N) tăng dần, X  
//output: dãy (a, N) đã có X ở đúng vị trí  
  
void insert(int mang[], int *N, int X)  
{  
    int pos= *N -1;  
    while (*N >= 0 && mang[pos]>X) {  
        mang[pos+1] = mang[pos];  
        pos--;  
    }  
    mang[pos+1] = X;  
    *N = *N + 1;  
}
```

Mảng – Loại bỏ một thành phần dữ liệu

- Bài toán: loại bỏ thành phần dữ liệu X ra khỏi mảng a đang có N thành phần.
- Hướng giải quyết: xác định vị trí của X, nếu tìm thấy thì dồn các phần tử ở phía sau lên để lấp vào chỗ trống. 2 trường hợp:
 - Dãy không có thứ tự: lấp phần tử cuối lên
 - Dãy đã có thứ tự: dời tất cả các phần tử ở sau vị trí của X lên trước 1 vị trí.

Mảng – Loại bỏ X ra khỏi dây tăng

Loại 5 khỏi (a, 8)



Tìm vị trí của 5

Dồn các vị trí 4, 5, 6, 7 lên

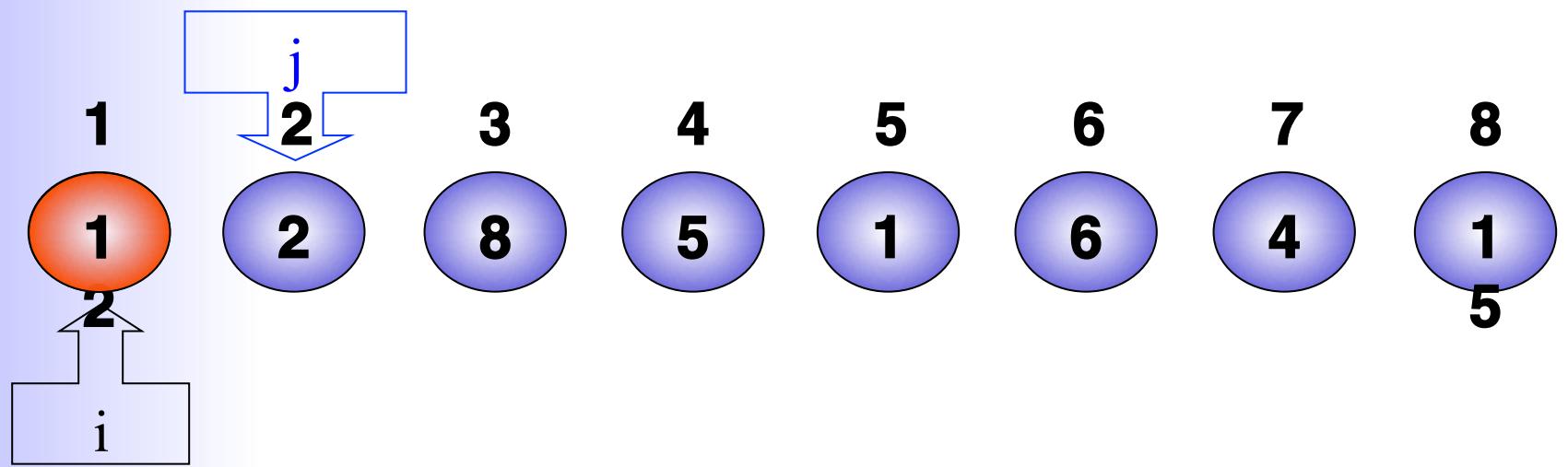
Mảng – Loại bỏ X ra khỏi dãy tăng

```
//input: dãy (a, N), X  
//output: dãy (a, N) đã loại bỏ 1 thành phần X  
  
int      Remove(int a[], int *N, int X)  
{  
    int pos = Search(a, *N, X);  
    if (pos == -1) //không có X trong dãy  
        return 0;  
    *N = *N-1;  
    for (; (pos < *N); pos++)  
        a[pos] = a[pos + 1];  
    return 1;  
}
```

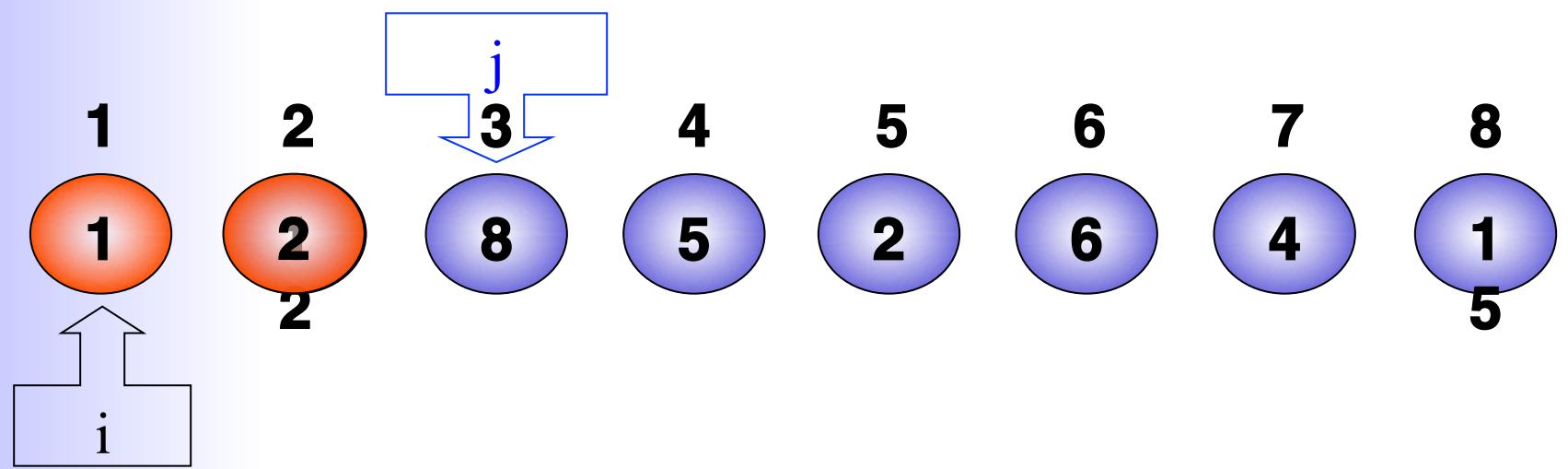
Mảng – Sắp xếp

- Bài toán: Sắp xếp các thành phần của (a, N) để thu được dãy tăng dần
- Giải pháp: Tìm cách loại bỏ tất cả các vị trí sai trong dãy
 - Thuật toán sắp xếp **Đôi chở trực tiếp**
 - Thuật toán nổi bọt - Bubble sort
 - Thuật toán Selection Sort
 - ...

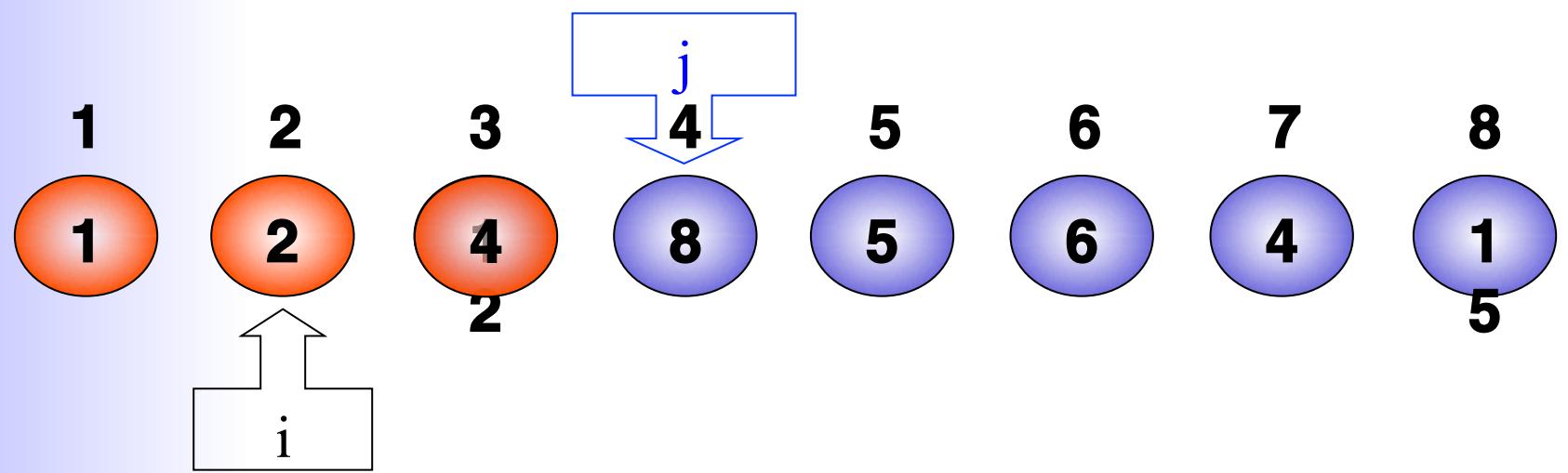
Mảng – Sắp xếp đổi chỗ



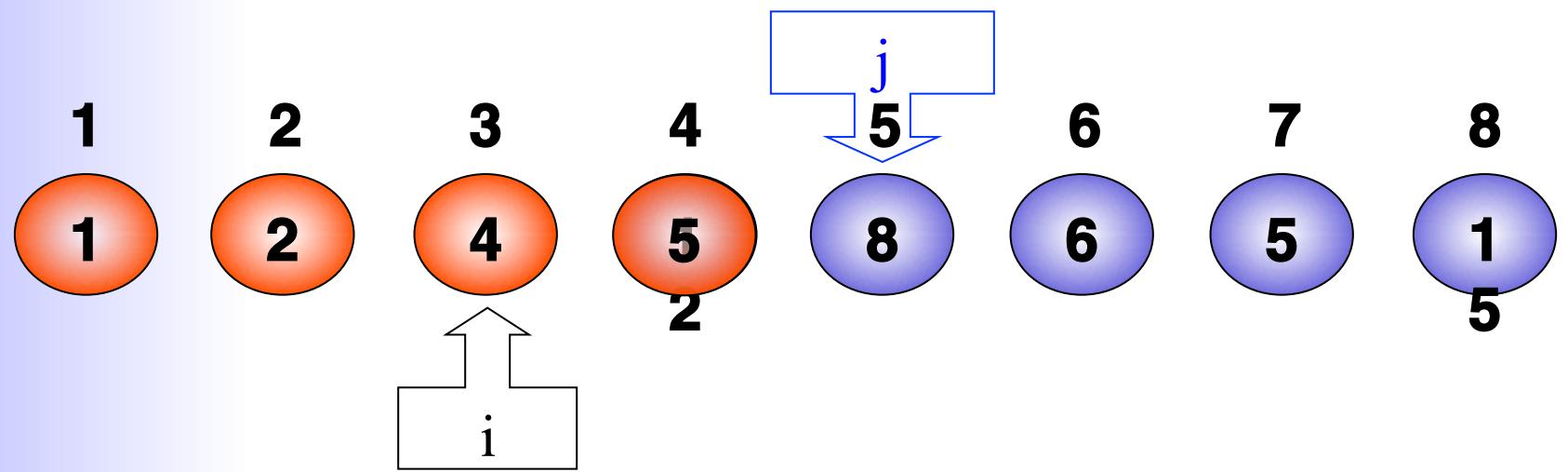
Mảng – Sắp xếp đổi chỗ



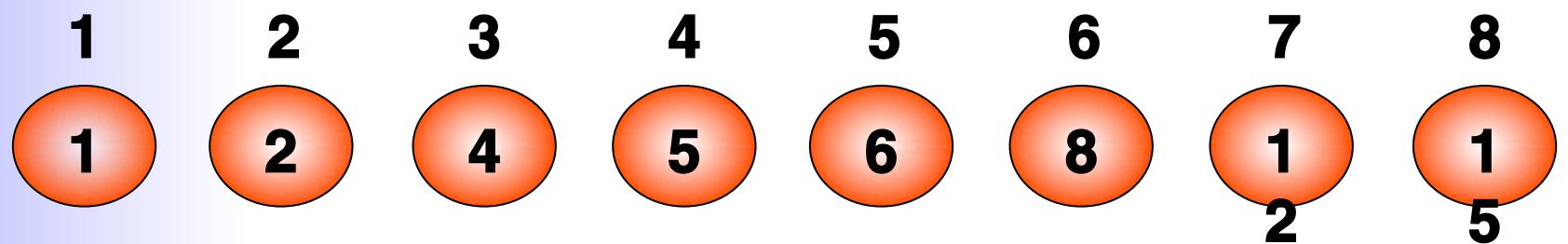
Mảng – Sắp xếp đổi chỗ



Mảng – Sắp xếp đổi chỗ



Mảng – Sắp xếp đổi chỗ



Mảng – Sắp xếp đổi chỗ

```
void Swap(int *x, int *y)
{
    int t = *x; *x = *y; *y = t;
}

void InterchangeSort(int a[], int N)
{
    int i, j;
    for (i = 0 ; i<N-1 ; i++)
        for (j =i+1; j < N ; j++)
            if(a[j]< a[i])
                Swap(&a[i],&a[j]);
}
```

Mảng nhiều chiều

- C không hỗ trợ mảng nhiều chiều. Tuy nhiên có thể tiếp cận theo hướng: Mảng 2 chiều là mảng một chiều mà mỗi thành phần của nó là một mảng một chiều.

```
float rainfall[12][365];
```

“rainfall” là mảng gồm 12 thành phần, mỗi thành phần là mảng gồm 365 số float

```
short exam_marks[500][10];
```

“exam_marks” là mảng gồm 500 thành phần, mỗi thành phần là mảng 10 số short

```
const int brighton = 7;  
int day_of_year = 238;
```

```
rainfall[brighton][day_of_year] = 0.0F;
```

Tổng kết

- Khai báo mảng trong C
- Truy xuất các phần tử
- Truyền tham số kiểu mảng cho hàm
- Các thao tác: nhập, xuất, thêm/hủy 1 thành phần, tìm kiếm, sắp xếp
- Mảng nhiều chiều