Mảng

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC MỞ TPHCM

Mục tiêu

- Biết cách sử dụng cấu trúc mảng để lưu trữ một tập hợp dữ liệu.
- Biết cách khai báo, khởi tạo và nhập/xuất từng giá trị trong mảng.
- Biết cách dùng câu lệnh lặp để truy cập đến từng giá trị trong mảng.
- Biết một số thao tác cơ bản trên mảng như tìm kiếm, tính toán với các giá trị dữ liệu, kiểm tra tính chất mảng, thêm/xóa các giá trị…
- Biết cách truyền mảng dữ liệu cho hàm.

Nội dung

1. Mảng
   - Khái niệm
   - Khai báo mảng
   - Khởi tạo mảng khi khai báo
   - Truy xuất các giá trị trong mảng
2. Các thao tác trên mảng
3. Một số lưu ý
4. Truyền mảng cho hàm

Khái niệm

- Mảng/Dãy (array) là một tập hợp gồm các phần tử có cùng kiểu dữ liệu, được lưu trữ tại các vị trí liên tục trong bộ nhớ.

[0]	8
[1]	12
[2]	23
[3]	0
[4]	14
[5]	25
[6]	7
[7]	10
[8]	5
[9]	16

- Ví dụ mảng số nguyên, mảng số thực, mảng ký tự...

- Mảng một chiều (one-dimensional array) là mảng có các phần tử được sắp xếp theo dạng danh sách (list).

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]
list	35	12	27	18	45	16	38	4	17	9	23

- Mảng hai chiều (two-dimensional array) là mảng có các phần tử được sắp xếp theo dạng bảng (table).

inStock	[RED]	[BROWN]	[BLACK]	[GRAY]	[WHITE]
[GM]	8	7	10	6	2
[FORD]	12	10	6	9	17
[TOYOTA]	12	5	10	4	8
[BMW]	8	16	6	3	2
[NISSAN]	14	11	13	9	3
[VOLVO]	7	8	9	4	12

Khai báo mảng

- Cú pháp khai báo mảng một chiều:

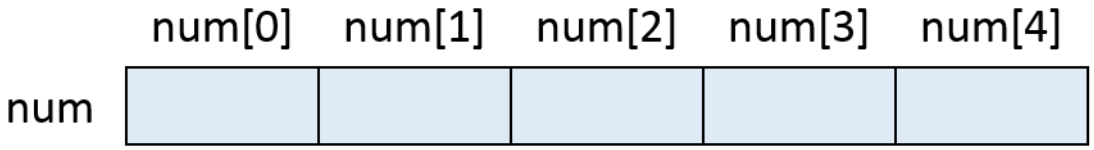
```
dataType arrayName[numberOfElements];
```

  - dataType: kiểu dữ liệu của mỗi phần tử trong mảng.
  - arrayName: tên của mảng.
  - numberOfElements: số nguyên, cho biết số phần tử (tối đa) của mảng.
- Số phần tử được xác định khi khai báo và không thay đổi.
- Các phần tử được đánh số từ 0 đến numberOfElements - 1

- Ví dụ khai báo mảng tên là num chứa 5 số nguyên:

```
int num[5];
```

  - 5 vị trí liên tục trong bộ nhớ được sử dụng để lưu 5 số nguyên. Nếu kiểu int có kích thước 4 bytes thì mảng sẽ chiếm 20 bytes.
  - Các phần tử gồm num[0], num[1], num[2], num[3], num[4].



- Khi khai báo phải biết trước số phần tử tối đa của mảng.
- Ví dụ khai báo sau đây gây lỗi:

```
int arrSize;

cout << "Nhap so phan tu cua mang: ";
cin >> arrSize;

int arr[arrSize]; //loi
```

- Nếu muốn chỉ định số phần tử lưu trong mảng vào lúc chương trình thực hiện thì dùng biến con trỏ (pointer) để cấp phát số phần tử của mảng, gọi là mảng cấp phát động (dynamic array).

- Định nghĩa số phần tử của mảng là hằng:

```
const int MAXSIZE = 5;
int num[MAXSIZE];   //tuong duong int num[5];
```

- Định nghĩa kiểu dữ liệu là mảng:

```
const int MAXSIZE = 50;
typedef double List[MAXSIZE];
List yourList;
List myList;
```

- Các phát biểu trên tương đương với:

```
double yourList[50];
double myList[50];
```

Khởi tạo mảng khi khai báo

- Mảng được khởi tạo giá trị khi khai báo theo dạng sau:

```
dataType arrayName[numberOfElements] = {initialValues};
```

  - initialValues: các giá trị phân cách bằng dấu phẩy (,).

- Ví dụ khai báo mảng gồm 3 phần tử và khởi tạo giá trị:

```
int a[3] = {2, 12, 1};
```

- tương đương:

```
int a[3];
a[0] = 2;
a[1] = 12;
a[2] = 1;
```

- Khi khai báo mảng không chỉ định kích thước nhưng khởi tạo bằng một danh sách các giá trị thì kích thước mảng được tính bằng cách đếm số phần tử trong danh sách khởi tạo.

```
double sales[5] = {12.25,32.50,16.90,23,45.68};
```

tương đương:

```
double sales[] = {12.25,32.50,16.90,23,45.68};
```

- Nếu danh sách khởi tạo có số phần tử ít hơn số phần tử của mảng thì các phần tử còn lại được khởi tạo là 0.
- Ví dụ khởi tạo tất cả các phần tử là 0.

```
int list[10] = {0};
```

- Ví dụ chỉ khởi tạo 3 phần tử, các phần tử còn lại là 0.

```
int list[10] = {8, 5, 12};
```

- tương đương:

```
int list[10] = {8, 5, 12, 0, 0, 0, 0, 0, 0, 0};
```

Truy xuất các giá trị trong mảng

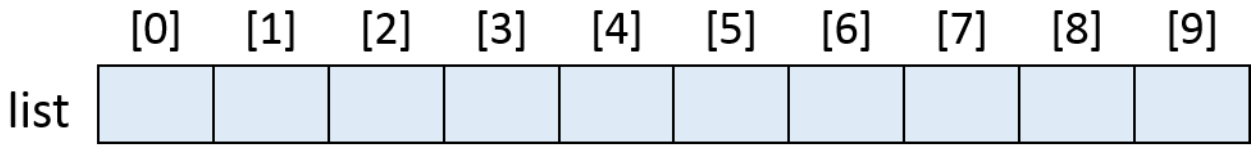
- Cú pháp gán dữ liệu cho một phần tử (element) trong mảng:

```
arrayName[subscript] = expression;
```

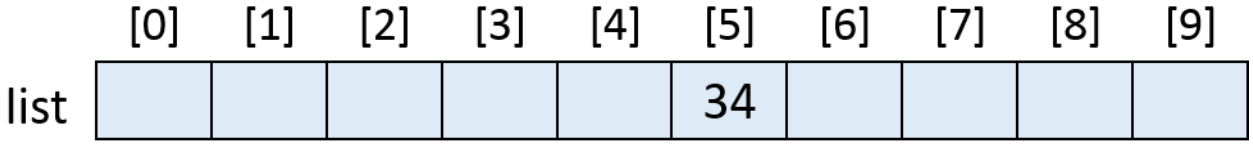
  - subscript: chỉ số là số nguyên hoặc biểu thức có giá trị là số nguyên. Nếu mảng có số phần tử là MAXSIZE thì chỉ số có giá trị từ 0 đến MAXSIZE - 1.
  - expression: biểu thức có kiểu phù hợp với kiểu dữ liệu của mảng.

- Ví dụ:

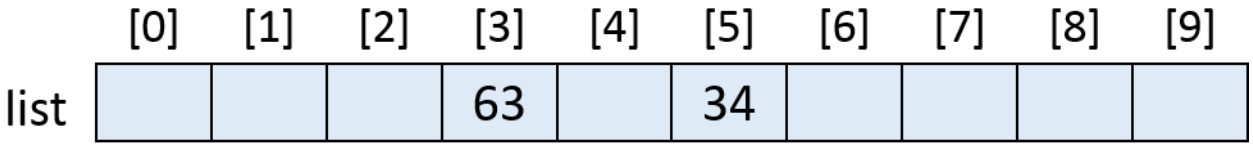
```
int list[10];
```



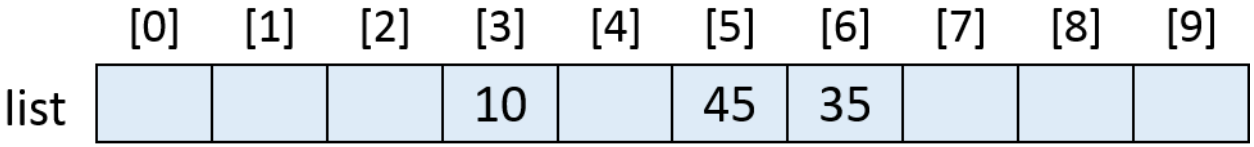
```
list[5] = 34;
```




```
int i;
i = 3;
list[i] = 63;  //tuong duong list[3] = 63;
```




```
list[3] = 10;
list[6] = 35;
list[5] = list[3] + list[6];
```



Nội dung

1. Mảng
2. Các thao tác trên mảng
   - Nhập/Xuất
   - Tính tổng và trung bình
   - Tìm kiếm
   - Tìm phần tử lớn nhất/nhỏ nhất
3. Một số lưu ý
4. Truyền mảng cho hàm

Các thao tác trên mảng

- Các thao tác thường gặp trên mảng gồm:
  - Nhập và in dữ liệu
  - Tìm một giá trị, tìm giá trị nhỏ nhất, lớn nhất
  - Tính tổng và giá trị trung bình
  - Thêm/Xóa
  - Sắp xếp tăng dần/giảm dần
  - ...
- Các thao tác này dùng vòng lặp để truy cập phần tử trong mảng.

Khởi tạo mảng

- Chương trình khởi tạo mảng gồm 10 số 0, sau đó in mảng.

```
#include <iostream>
using namespace std;

int main()
{
    const int MAXSIZE = 10;
    double a[MAXSIZE];

    for (int i = 0; i <= MAXSIZE - 1; i++)
        a[i] = 0;

    for (int i = 0; i <= MAXSIZE - 1; i++)
        cout << a[i] << " ";
    cout << endl;
    return 0;
}
```

Nhập/Xuất

- Chương trình nhập mảng gồm 10 số, sau đó in ra mảng.

```
#include <iostream>
using namespace std;

int main()
{
    const int MAXSIZE = 10;
    double a[MAXSIZE];

    cout << "Nhap " << MAXSIZE << " so double: ";
    for (int i = 0; i <= MAXSIZE - 1; i++)
        cin >> a[i];

    for (int i = 0; i <= MAXSIZE - 1; i++)
        cout << a[i] << " ";
    cout << endl;
}
```

Tính tổng và trung bình

- Chương trình tính tổng và trung bình các phần tử trong mảng.

```
#include <iostream>
using namespace std;

int main()
{
    double a[] = { 8, 6, 7, 1, 4, 2, 3, 10, 9, 5 };
    int n = sizeof(a) / sizeof(a[0]);
    cout << "\nCac gia tri trong mang:" << endl;
    for (int i = 0; i <= n - 1; i++)
        cout << a[i] << " ";
    double tong = 0;
    for (int i = 0; i <= n - 1; i++)
        tong += a[i];
    cout << "\nTong = " << tong;
    cout << "\nTrung binh = " << tong / n;
}
```

Tìm kiếm

- Chương trình tìm một giá trị trong mảng và in ra vị trí tìm thấy.
- Ví dụ tìm 27 trong mảng gồm có 7 số nguyên:

	[0]	[1]	[2]	[3]	[4]	[5]	[6]
a	35	12	27	18	45	16	38

- Giải pháp: Tìm từ đầu cho đến cuối mảng:
  - So sánh 27 với a[0], a[1], a[2], ..., a[6].
  - Do a[2] == 27 nên quá trình tìm kiếm thành công, trả về vị trí 2.

- Thuật giải tìm x trong mảng a có n phần tử:
  1. vitri = 0
  2. while (vitri <= n-1 and a[vitri] != x)
  3. vitri = vitri + 1
  4. if (vitri <= n-1)
  5. print "tim thay tai vitri"
  6. else print "khong tim thay"

```cpp
#include <iostream>
using namespace std;

int main()
{
    int a[] = { 35, 12, 27, 18, 45, 16, 38 };
    int n = sizeof(a) / sizeof(a[0]);
    int x, vitri;
    cout << "Nhap so can tim: ";
    cin >> x;
    vitri = 0;
    while (vitri <= n - 1 && a[vitri] != x)
        vitri++;
    if (vitri <= n - 1)
        cout << "Tim thay tai vi tri " << vitri << endl;
    else
        cout << "Khong tim thay" << endl;
}
```

Tìm phần tử lớn nhất

- Chương trình tìm và in ra phần tử lớn nhất trong mảng.
- Ví dụ tìm giá trị lớn nhất trong mảng sau đây:

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
a	12.5	8.35	19.60	25.00	14.00	39.43	35.90	98.23	66.65	35.64

- Giải pháp:
  - lonnhat = a[0]
  - So sánh lonnhat với a[1],..., a[n-1] và cập nhật lại lonnhat

i	lonnhat	a[i]	lonnhat < a[i]
1	12.50	8.35	false
2	12.50	19.60	true
3	19.60	25.00	true
...	...	...	...
7	39.43	98.23	true
8	98.23	66.65	false
9	98.23	35.64	false

```
#include <iostream>
using namespace std;

int main()
{
    double a[] = { 12.5,8.35,19.6,25,14,39.43,35.9,
                   98.23,66.65,35.64 };
    int n = sizeof(a) / sizeof(a[0]);

    double lonnhat = a[0];
    for (int i = 1; i <= n - 1; i++)
        if (lonnhat < a[i])
            lonnhat = a[i];

    cout << "Gia tri lon nhat = " << lonnhat << endl;
    return 0;
}
```

Nội dung

1. Mảng
2. Các thao tác trên mảng
3. Một số lưu ý
4. Truyền mảng cho hàm

Một số lưu ý

- Không truy xuất phần tử ngoài phạm vi mảng.
- Ví dụ:

```
int a[10];

for (int i = 0; i <= 10; i++)
    a[i] = 0;
```

  - các chỉ số i hợp lệ là 0, 1, 2,..., 9
  - khi i = 10, vòng lặp kiểm tra điều kiện i <= 10 đúng và gán 0 vào a[10], nhưng a[10] không tồn tại!

- Không dùng phép gán một mảng cho một mảng khác.
- Ví dụ:

```
int a[5] = {0, 4, 8, 12, 16};
int b[5];
b = a; //illegal
```

- Dùng vòng lặp để sao chép các giá trị của mảng:

```
for (int i = 0; i < 5; i ++)
    b[i] = a[i];
```

- Không dùng cin/cout với biến mảng.
- Ví dụ:

```
int a[10]; int b[10];
cin >> a;
//...    //illegal
cout << b;     //illegal
```

- Dùng vòng lặp để nhập/xuất dữ liệu trong mảng:

```
for (int i = 0; i < 10; i ++)
    cin >> a[i];
```

- Không dùng phép so sánh đối với biến mảng:

```
if (a < b)     //illegal
    //...
```

Nội dung

1. Mảng
2. Các thao tác trên mảng
3. Một số lưu ý
4. Truyền mảng cho hàm

Truyền mảng cho hàm

- Trong C++, biến mảng luôn được truyền cho hàm bằng tham chiếu (pass-by-reference).
  - Không cần dùng ký hiệu & khi khai báo tham số là mảng.
- Khi tham số của hàm là mảng một chiều:
  - không cần khai báo số phần tử tối đa của mảng;
  - nhưng phải có tham số cho biết số phần tử hiện có trong mảng.
- Ví dụ hàm khởi tạo các giá trị ban đầu cho mảng:

```
void khoitao(int arr[], int arrSize)
{
    for (int i = 0; i <= arrSize - 1; i++)
        arr[i] = 0;
}
```

- Khi truyền tham chiếu, hàm có thể thay đổi giá trị của tham số. Vì vậy, để ngăn không cho hàm thay thay đổi giá trị của tham số mảng, ta dùng tham chiếu hằng.

```
void xuat(const int arr[], int arrSize)
{
    for (int i = 0; i <= arrSize - 1; i++)
        cout << arr[i] << " ";
    cout << endl;
}
```

- Đây là một kỹ thuật lập trình tốt: khai báo tham số mảng là tham chiếu hằng nếu hàm không hiệu chỉnh các giá trị lưu trong mảng.
- C++ không cho phép hàm trả về giá trị có kiểu mảng.
- Ví dụ hàm tính giá trị trung bình của một mảng số thực.

```
#include <iostream>
using namespace std;

double tinhTrungbinh(double arr[], int arrSize)
{
    double tong = 0;
    for (int i = 0; i <= arrSize - 1; i++)
        tong += arr[i];
    return tong / arrsize;
}

int main()
{
    double a[] = { 1,2,3,4,5,6,7,8,9,10 };
    int n = sizeof(a) / sizeof(a[0]);
    cout << "Trung binh = " << tinhTrungbinh(a, n)
         << endl;
}
```

Q & A