LẬP TRÌNH NÂNG CAO

Bài 4+5+6: Kiểu dữ liệu mảng và xâu ký tự trong C/C++

Nội dung chính

1. Kiểu dữ liệu mảng
    1. Khái niệm và khai báo
    2. Mảng nhiều chiều
    3. Mảng vs Vector
    4. Hàm với tham số kiểu mảng
    5. Vòng lặp phạm vi
    6. Các bài toán cơ bản với kiểu mảng
2. Kiểu xâu kí tự
    1. Khái niệm và khai báo
    2. Các phép toán trên xâu kí tự
    3. Các bài toán cơ bản với kiểu xâu kí tự
    4. Xâu kí tự vs Chuỗi (string)
3. Bài tập

Phần 1

Kiểu dữ liệu mảng

1.1 Khái niệm và khai báo

- Mảng = Dãy các biến cùng kiểu, cùng tên, khác chỉ số
  - Chỉ số là số tự nhiên, luôn bắt đầu từ 0
  - Là giải pháp cho phép lưu trữ một dãy các biến tương đương, thay vì phải chỉ ra từng biến một
  - Vay mượn cảm hứng từ dãy số trong toán học
- Ví dụ:

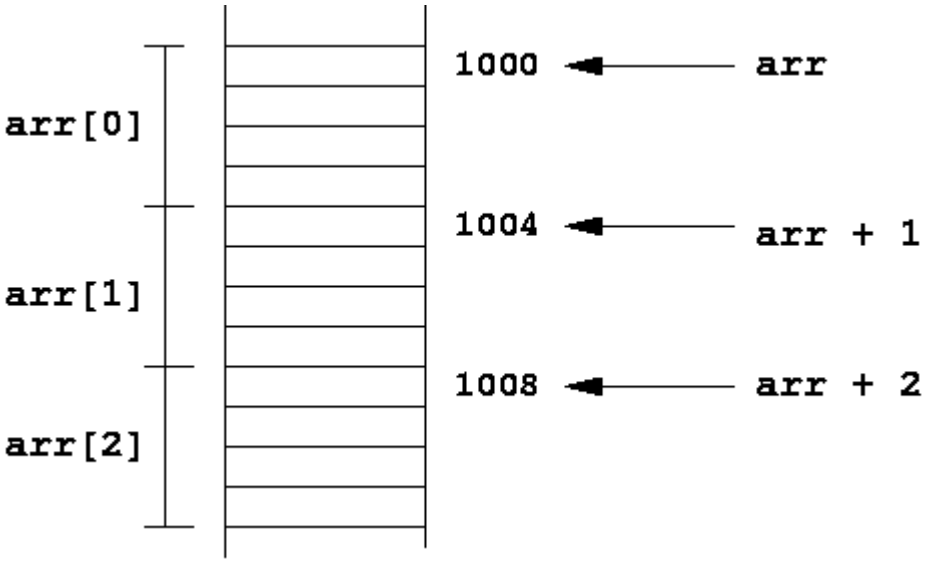
```
double a[10];                          // mảng 10 số thực
int b[] = { 1, 2, 3 };                 // mảng 3 số nguyên
int c[4] = { 1, 2, 3 };                // mảng 4 số nguyên
bool d[2] = { true, false, true };     // lỗi
```

- Bản chất là một dãy biến:

```
int c[4] = { 1, 2, 3 };
int c[0] = 1, c[1] = 2, c[2] = 3, c[3] = ?;
```

1.1 Khái niệm và khai báo

- Quy tắc:
  - Tên mảng quy tắc đặt như tên biến
  - Kích cỡ (số phần tử) được xác định ngay khi khai báo (thường phải là hằng số)*
  - Kích cỡ không thể thay đổi
  - Sẽ là một khối nhớ liên tục chứa các biến



1.1 Khái niệm và khai báo: một số lỗi hay gặp

- Khai báo không chỉ rõ số lượng phần tử
  - `int a[]; => int a[100];`
- Số lượng phần tử liên quan đến biến hoặc hằng
  - `int n1 = 10; int a[n1]; => int a[10];`
  - `const int n2 = 10; int a[n2]; => int a[10];`
- Khởi tạo cách biệt với khai báo
  - `int a[4]; a = {2912, 1706, 1506, 1904};`
  - `=> int a[4] = {2912, 1706, 1506, 1904};`
- Chỉ số mảng không hợp lệ
  - `int a[4];`
  - `a[-1] = 1; a[10] = 0;`

1.2 Mảng nhiều chiều

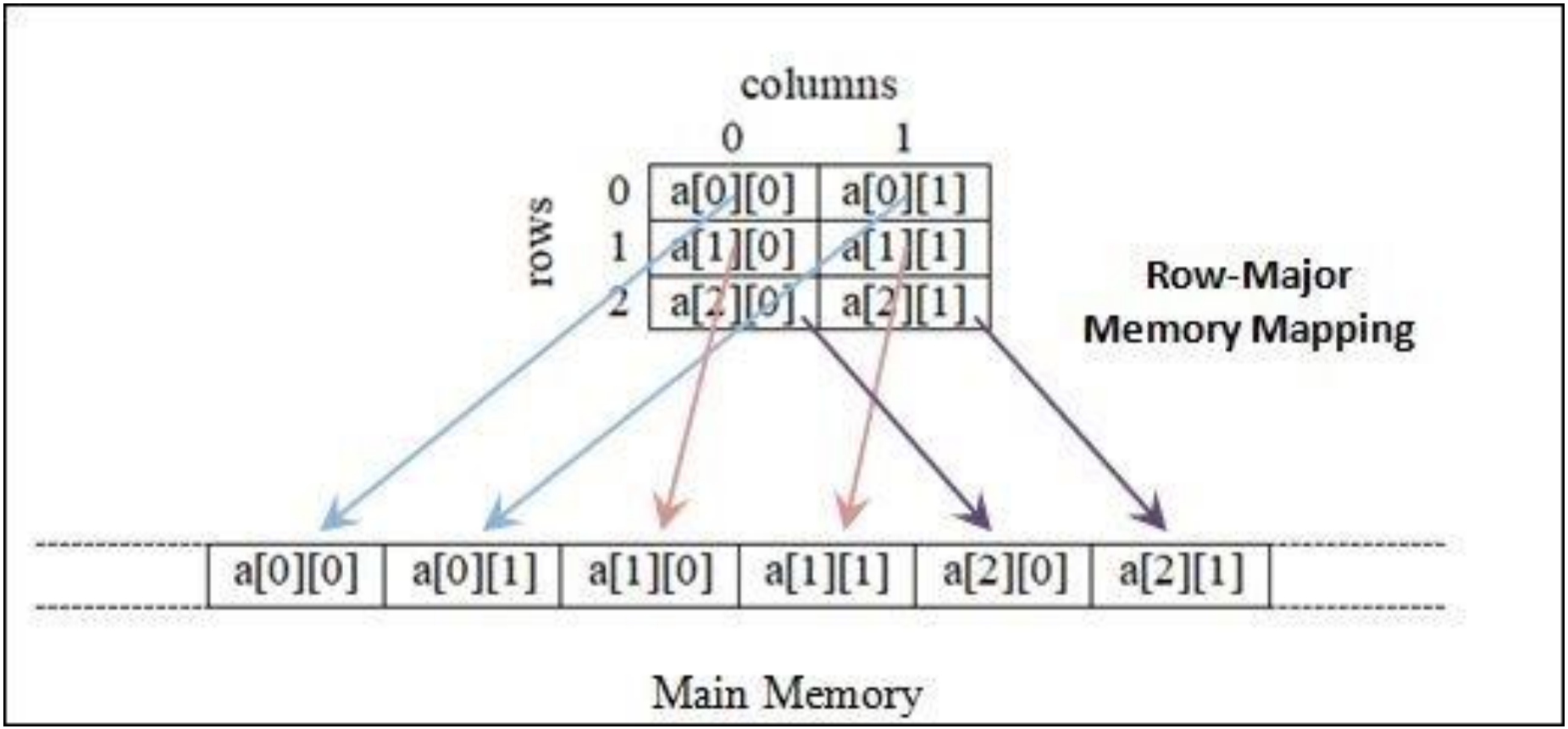
- Mảng nhiều chiều = nhiều mảng con giống nhau
- Ví dụ:

```
// mảng số thực hai chiều 3 hàng x 5 cột
// ~ 3 mảng một chiều 5 số thực
double mang21[3][5];

// mảng số nguyên hai chiều 3 hàng x 4 cột
// chú ý cách khởi tạo dữ liệu
int mang22[3][4] = {
    { 1, 2,  3, 4 },
    { 5, 6,  7 },
    { 8, 9, 10 }
};
```

1.2 Mảng nhiều chiều

- Cũng như mảng một chiều, trình dịch C/C++ bố trí các biến nằm liên tiếp thành một khối trong bộ nhớ

```
int mang22[3][4] = { 1, 2, 3, 4, 5, 6 };
```



1.3 Mảng vs Vector

Mảng

- Cú pháp đơn giản
- Không cần thư viện ngoài
- Số phần tử cố định
- Không thể có mảng 0 phần tử
- Không có hàm bổ trợ
- Không thể trả về từ hàm
- Không an toàn khi sử dụng trong hàm (thay đổi giá trị)
- Không thể gán cho nhau
- Phải gán giá trị từng phần tử khi sao chép mảng

Vector

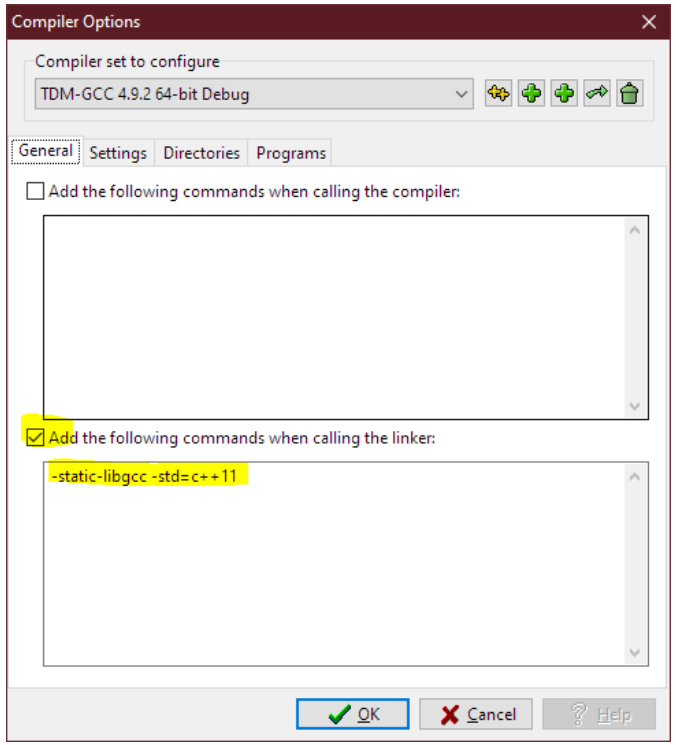
- Cú pháp phức tạp hơn
- Cần thư viện bên ngoài
- Có thể thay đổi số phần tử
- Có thể có 0 phần tử
- Nhiều phương thức hỗ trợ
- Có thể trả về từ hàm
- An toàn khi sử dụng trong hàm (không thay đổi giá trị)
- Có thể gán cho nhau
- Dùng phép gán biến khi sao chép vector

1.4 Hàm với tham số kiểu mảng

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

typedef int Mang[100];          // định nghĩa kiểu mảng

void change(Mang x) {
    x[0] = 10;
}

int main() {
    Mang a = { 3, 2, 1 };       // khởi tạo 3 phần tử đầu
    cout << a[0] << endl;       // in ra 3
    change(a);                  // đổi a[0] thành 10
    cout << a[0] << endl;       // in ra 10
}
```

1.5 Vòng lặp phạm vi

- Đây là cú pháp xuất hiện từ bản C++11 (2011)
- Hầu hết các trình dịch C/C++ mới đều hỗ trợ cú pháp này
- Hiện Dev-C++ đang dùng C++98, có chỉnh phiên bản lên mới bằng option sau: menu => Tools => Compiler Options
- Gõ vào nội dung dòng "-static-libgcc -std=c++11"
- Bật đánh dấu tick như hình bên



1.5 Vòng lặp phạm vi

- Tiếng Anh: range-base for loop
- Một cấu trúc for mới
- Nhiều ngôn ngữ đã có (gọi là vòng foreach)
- Ví dụ: in phần tử của mảng một chiều

```
int a[5] = { 1, 2, 3, 4, 5 };    // mảng a
for (int i : a)                  // duyệt các phần tử
    cout << i;                   // in ra từng phần tử
```

- Ví dụ: in phần tử của mảng nhiều chiều

```
double mang21[3][5];
for (auto & x : mang21) {
    for (auto & y : x) cout << y << " ";
    cout << endl;
}
```

1.6 Các bài toán cơ bản với kiểu mảng

- Nhiều dạng, sử dụng rất nhiều trong cuộc sống
  - Danh sách điểm số, sinh viên,... => mảng 1 chiều
  - Âm thanh số hóa => mảng 1 chiều
  - Hình ảnh số hóa => mảng 2 chiều
  - Dữ liệu tài nguyên, đất đai, không gian,... => mảng 3 chiều
- Nhập / xuất dữ liệu kiểu mảng
- Tìm kiếm theo đặc trưng
- Tìm kiếm nhanh
- Sắp xếp theo tiêu chí
- Thay thế, tính toán trên dữ liệu
- Sao chép, ghép nối, thống kê...

1.6 Các bài toán cơ bản với kiểu mảng

- Thư viện <algorithm> cung cấp một số thường dùng cho các vấn đề cơ bản
- Với dãy thông thường
  - sort: sắp xếp một dãy
  - find: tìm kiếm trong dãy
- Với dãy đã sắp tăng dần
  - binary_search: kiểm tra xem có phần tử trong đoạn tăng dần hay không
  - lower_bound: trả về vị trí của phần tử đầu tiên không bé hơn phần tử cần tìm
  - upper_bound: trả về vị trí của phần tử đầu tiên lớn hơn phần tử cần tìm

Phần 2

Kiểu xâu kí tự

2.1 Khái niệm và khai báo

- Một kí tự:
  - 1 byte, char (-128...127) hoặc unsigned char (0...255)
  - Kiểu dữ liệu số nguyên
  - Viết trong cặp ngoặc đơn: `'a' 'x' '&'`
  - Hoặc viết giá trị mã của chữ
  - Tức là hai cách viết dưới đây hoàn toàn như nhau:

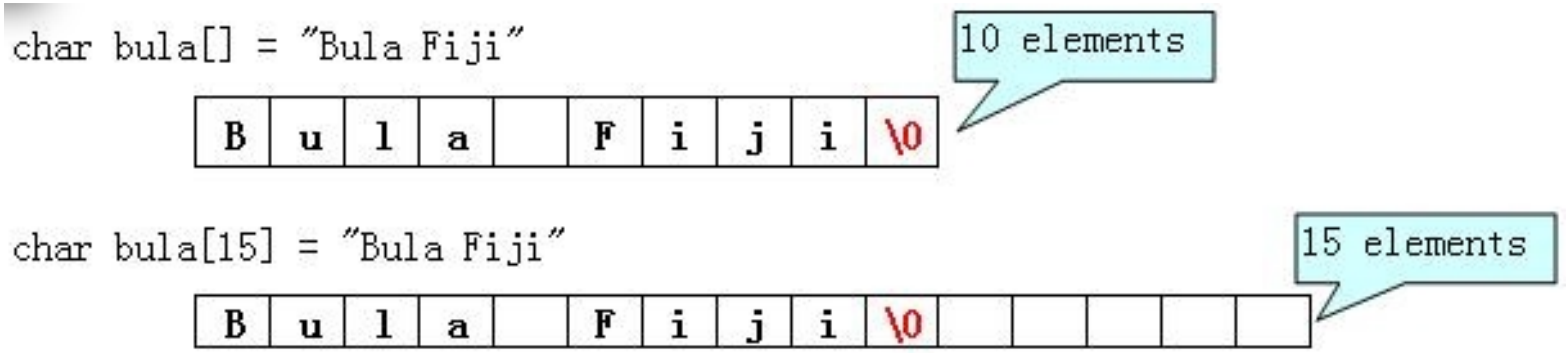
```
char a = 97;
char a = 'a';
```

  - Hạn chế: mã hóa mã ASCII
  - Trong các phiên bản mới C/C++ có kiểu dữ liệu wchar để mã hóa unicode
- Mảng kí tự: như mảng các số nguyên

```
char a[3] = {'a','b', 70 };
```

2.1 Khái niệm và khai báo

- Xâu kí tự = dãy kí tự + kí tự 0 kết thúc



- Đây là quy tắc đánh dấu của C/C++:
  - Xâu kí tự kết thúc bởi kí tự 0 (kí tự có mã 0 ≠ số 0)
  - Độ dài của xâu kí tự không tính kí tự 0 này
  - Bộ nhớ để lưu xâu kí tự thì cần
- Thử

```
char a[3] = {'a','b', 70};
a[1] = 0;
cout << a;        // chỉ in ra 'a'
```

2.1 Khái niệm và khai báo

- Các cách khai báo dưới đây là như nhau:

```
char str[4] = "C++";
char str[]  = {'C','+','+','\0'};
char str[4] = {'C','+','+','\0’};
```

- Kiểu `size_t = unsigned int`, là kiểu dữ liệu chuyên dùng để biểu diễn kích cỡ khi làm việc với bộ nhớ và kích cỡ các biến
- Phép toán `sizeof(X)`: trả về kích cỡ của X
  - X có thể là một biểu thức

2.2 Các phép toán trên xâu kí tự

- Thư viện <cstring> (string.h): nhiều hàm kiểu xâu kí tự
  - strlen
  - strcpy
  - strdup
  - strlwr/strupr
  - strrev
  - strcmp/stricmp
  - strcat
  - strstr
- Tra cứu: http://www.cplusplus.com/reference/cstring/
- Các hàm của cstring sử dụng con trỏ (sẽ được học ở phần sau): `char array ≈ const char *`

2.3 Các bài toán cơ bản với kiểu xâu kí tự

- Nhập dữ liệu
  - `cin.get(str, max_length);`
- Xuất dữ liệu
- Xử lý dữ liệu chuỗi:
  - Tìm kiếm
    - Tìm kiếm chính xác
    - Tìm kiếm gần đúng
  - So sách
    - So sánh chính xác
    - So sánh bỏ qua những đặc trưng ngôn ngữ
    - So sánh gần đúng
  - Thống kê
  - Phát hiện tương quan

2.4 Xâu kí tự vs Chuỗi (string)

Xâu kí tự

- Kiểu dữ liệu sẵn có của C/C++
- Cú pháp tương đối đơn giản
- Không cần thư viện ngoài
- Hàm bổ trợ dùng con trỏ
- Không thể trả về từ hàm
- Không an toàn khi sử dụng trong hàm (thay đổi giá trị)
- Không thể gán cho nhau
- Phải gán giá trị từng phần tử khi sao chép mảng

Chuỗi

- Kiểu dữ liệu của thư viện std
- Cú pháp tương đối đơn giản
- Cần thư viện bên ngoài
- Nhiều phương thức hỗ trợ
- Có thể trả về từ hàm
- An toàn khi sử dụng trong hàm (không thay đổi giá trị)
- Có thể gán cho nhau
- Dùng phép gán biến khi sao chép string

Phần 3

Bài tập

Bài tập về mảng

1. Viết hàm lonhon nhận đầu vào là mảng a, số phần tử n và số thực d, hàm trả về số lượng phần tử lớn hơn d
2. Viết hàm solonhon nhận đầu vào là mảng a, số phần tử n và số thực d, hàm trả về vị trí đầu tiên của phần tử lớn hơn d trong mảng a, trả về -1 nếu không có
3. Viết hàm phantich nhận đầu vào là mảng a, số phần tử n và số m, hàm trả về true nếu m có thể phân tích thành tổng 3 số trong a, ngược lại trả về false.
4. Có thể cắt dãy bất kỳ thành các dãy con tăng dần, ví dụ (1,3,2,4,5) = (1,3), (2,4,5); hoặc (1,3), (2,4), (5). Viết hàm chianho nhận đầu vào là mảng a và số phần tử n. Sau đó in ra cách cắt có ít dãy con nhất.

Bài tập về xâu kí tự

1. Tìm hiểu thêm một số hàm khác như:
   - Đối xâu thành số: atoi, atol, atof
   - Đối số thành xâu: itoa, ltoa, ultoa
   - Tách xâu: strtok
2. Viết các hàm nhận vào một xâu kí tự và trả về xâu kí tự tương ứng (không thay đổi chuỗi ban đầu):
   - Các kí tự thành kí tự thường (giống strlwr)
   - Các kí tự thành kí tự hoa (giống strupr)
   - Các kí tự đầu tiên mỗi từ thành kí tự hoa
   - Xóa khoảng trắng thừa ở đầu và cuối xâu (khoảng trắng được hình thành bởi kí tự space – 32 hoặc tab – 9)
   - Chuẩn hóa chuỗi (xóa khoảng trắng thừa)
   - Xóa các kí tự không phải chữ số, gạch dưới, chấm và phẩy

Bài tập về xâu kí tự

1. Viết hàm wordcount nhận vào một xâu kí tự s và trả về số từ trong xâu kí tự đó.
2. Viết hàm wordmax nhận vào một xâu kí tự s và trả về độ dài từ có chiều dài lớn nhất đồng thời in nội dung từ đó ra màn hình.
3. Viết hàm sort2 nhận vào một xâu kí tự s, trong xâu s chỉ có các chữ cái và chữ số, hàm sắp xếp lại nội dung của s sao cho các chữ cái dồn hết về đầu chuỗi và các chữ số dồn hết về cuối chuỗi nhưng giữ nguyên thứ tự tương đối giữa chúng.
   Ví dụ: "k1j2h31k" => "1231kjhk"