Chương 3: Mảng, Chuỗi và con trỏ

Phần 1: Lý thuyết:

1. Mảng (Array)

Mảng 1 chiều:

Khai báo và khởi tạoint numbers[5] = {1, 2, 3, 4, 5};
numbers[0] = 10; Thay đổi phần tử đầu tiên

Truy cập phần tử: numbers[index] (index từ 0 đến size-1).
Kích thước cố định: Không thể thay đổi sau khi khai báo.
Mảng 2 chiều:

```
int matrix[2][3] = {
  {1, 2, 3},
  {4, 5, 6}};
cout << matrix[1][2];  Output: 6
```

---

2. Chuỗi (String)

Chuỗi C-style (char[]):

```
char name[20] = "John";  Khai báo với kích thước tối đa
```

cout << name; Output: John

Thư viện <cstring>: Hỗ trợ các hàm như strlen(), strcpy(), strcmp().
Kết thúc bằng ký tự \0: Đảm bảo không truy cập vượt quá giới hạn.
Lớp string (C++):

```
#include <string>
string greeting = "Hello";
greeting += " World!";  Nối chuỗi
cout << greeting.length();  Output: 11
```

---

3. Con Trỏ (Pointer)

Khai báo và sử dụng:

```
int value = 10;int *ptr = &value;  Con trỏ trỏ đến địa chỉ của biến value
cout << *ptr;  Output: 10 (Giá trị tại địa chỉ ptr)
```

Con trỏ và mảng:

```
int arr[3] = {1, 2, 3};int *ptr = arr;  ptr trỏ đến phần tử đầu tiên (arr[0])
cout << *(ptr + 1);  Output: 2 (arr[1])
```

Toán tử & và *:

&: Lấy địa chỉ của biến.

*: Truy cập giá trị tại địa chỉ con trỏ.

Phần 2: Thực hành

Bài 1: Sắp Xếp Mảng (Bubble Sort)

Mục tiêu: Sắp xếp mảng tăng dần.

Code mẫu:

```
void bubbleSort(int arr[], int size) {
  for (int i = 0; i < size-1; i++) {
    for (int j = 0; j < size-i-1; j++) {
      if (arr[j] > arr[j+1]) {
         Hoán đổi giá trị
        int temp = arr[j];
        arr[j] = arr[j+1];
        arr[j+1] = temp;
      }
    }
  }}
int main() {
  int arr[] = {5, 3, 8, 1};
  int size = 4;
  bubbleSort(arr, size);
  for (int i = 0; i < size; i++) {
    cout << arr[i] << " ";   Output: 1 3 5 8
  }
  return 0;}
```

Bài 2: Tìm Phần Tử Lớn Nhất Trong Mảng

Code mẫu:

```
int findMax(int arr[], int size) {
  int max = arr[0];
  for (int i = 1; i < size; i++) {
    if (arr[i] > max) max = arr[i];
  }
  return max;}
int main() {
  int arr[] = {10, 5, 20, 15};
  cout << "Max: " << findMax(arr, 4);   Output: 20
```

```
  return 0;}
```

Bài 3: Đảo Ngược Chuỗi

Cách 1: Dùng string:

```
string reverseString(string s) {
  int left = 0, right = s.length() - 1;
  while (left < right) {
    swap(s[left], s[right]);
    left++;
    right--;
  }
  return s;}
```

Cách 2: Dùng char[]:

```
void reverseCharArray(char str[]) {
  int len = strlen(str);
  for (int i = 0; i < len/2; i++) {
    swap(str[i], str[len - i - 1]);
  }}
```

Bài 4: Đếm Số Lần Xuất Hiện Của Ký Tự

Code mẫu:

```
int countChar(string s, char target) {
  int count = 0;
  for (char c : s) {
    if (c == target) count++;
  }
  return count;}
int main() {
  string text = "programming";
  cout << "So lan xuat hien 'g': " << countChar(text, 'g');   Output: 2
  return 0;}
```

Bài 5: Con Trỏ và Mảng

Ví dụ: Duyệt mảng bằng con trỏ:

```
int main() {
```

```
int arr[3] = {10, 20, 30};
int *ptr = arr;   ptr trỏ đến arr[0]
for (int i = 0; i < 3; i++) {
  cout << *(ptr + i) << " ";   Output: 10 20 30
}
return 0;}
```

Lỗi Thường Gặp & Cách Khắc Phục

Truy cập mảng ngoài giới hạn:
int arr[3] = {1, 2, 3};

cout << arr[3];   Lỗi! arr chỉ có index 0-2

→ Luôn kiểm tra index hợp lệ.
Quên kết thúc \0 trong chuỗi C-style:
char str[5] = "Hello";   Sai (thiếu chỗ cho \0)

→ Khai báo kích thước lớn hơn độ dài chuỗi.
Con trỏ chưa khởi tạo:
int *ptr;   Con trỏ "wild"*ptr = 5;   Lỗi!

→ Luôn gán địa chỉ hợp lệ cho con trỏ.

Tips Tối Ưu Hoá

Dùng string thay vì char[]: Tránh lỗi kích thước và dễ thao tác.
Con trỏ và mảng: Hiểu rằng arr[i] tương đương *(arr + i).
Sử dụng debugger: Xem giá trị mảng và con trỏ trong từng bước (VS Code > Run and Debug).