Bài 64. Mối quan hệ giữa con trỏ và mảng

Bởi **Nguyễn Văn Hiếu** -

Bài số 62 trong 69 bài của khóa học Học C Không Khó

Trong bài học này, Lập trình không khó sẽ cùng các bạn đi tìm hiểu **mối quan hệ giữa con trỏ và mảng** trong ngôn ngữ lập trình C. Bạn sẽ học thêm về một số toán tử của con trỏ, sử dụng các toán tử đó để duyệt mảng. Do đó, bạn sẽ biết thêm 1 cách mới để lặp qua mảng sử dụng con trỏ. Tất nhiên, mục tiêu cao hơn hết là giúp bạn hiểu sâu hơn, biết thêm các kiến thức về con trỏ trong ngôn ngữ C.

Trước khi bạn bắt đầu bài học này, bạn cần chắc chắn mình nắm rõ các kiến thức dưới đây:

- Mảng trong ngôn ngữ lập trình C
- Con trỏ trong ngôn ngữ lập trình C

NỘI DUNG BÀI VIẾT

Các phần tử của mảng là các ô nhớ liên tiếp

Nhắc lại khái niệm về mảng: "*Mảng là một tập hợp tuần tự các phần tử có cùng kiểu dữ liệu và các phần tử được lưu trữ trong một dãy **các ô nhớ liên tục** trên bộ nhớ*".

Các bạn đặc biệt lưu ý tới tính chất được lưu trên các ô nhớ liên tục, bây giờ chúng ta sẽ chứng minh tính đúng đắn của nó bằng ví dụ dưới đây:

```
0
1 #include <stdio.h>
```

```
2
3  int main(){
4      // Khai báo mảng có 5 phần tử
5      int arr[] = {1, 2, 3, 4, 5};
6
7      printf("Dia chi cua mang arr = %d\n", &arr);
8      printf("Gia chi cua mang arr = %d\n", arr);
9      // Thử in địa chỉ của từng phần tử
10     // sizeof (arr): Kích thước của mảng
11     // sizeof (int): Kích thước của kiểu int
12     for(int i = 0; i < sizeof (arr) / sizeof (int); i++){
13         printf("Dia chi cua arr[%d] = %d\n", i, &arr[i]);
14     }
15
16 }
17
```

Kết quả chạy chương trình trên:

```
0
1  Dia chi cua mang arr = 6487552
2  Gia chi cua mang arr = 6487552
3  Dia chi cua arr[0] = 6487552
4  Dia chi cua arr[1] = 6487556
5  Dia chi cua arr[2] = 6487560
6  Dia chi cua arr[3] = 6487564
7  Dia chi cua arr[4] = 6487568
8
```

Nhận xét:

- Các phần tử liên tiếp có địa chỉ cách nhau 4 giá trị, bởi vì 1 phần tử kiểu `int` có kích thước 4 bytes (máy tính x64). Nên ta chắc chắn các phần tử mảng được xếp cạnh nhau trong bộ nhớ.
- Một điều đặc biệt nữa, như mình có nói là khi truyền mảng vào hàm thì mặc định là truyền theo tham chiếu. Và trong ví dụ này bạn thấy đó, địa chỉ của biến mảng chính là địa chỉ của phần tử đầu tiên của mảng. Và giá trị của biến mảng cũng chính là địa chỉ của phần tử đầu tiên của mảng.
- Như vậy, `&arr[0]` tương đương `&arr` và tương đương `arr`. Điều đó có được là do biến `arr` trỏ tới phần tử đầu tiên của mảng.

Toán tử tăng và giảm của con trỏ

Giống như biến thông thường, con trỏ cũng có toán tử tăng và giảm. Nhưng cách toán tử tăng/ giảm trên con trỏ làm việc như nào.

```
0
#include <stdio.h>

int main()
{
    // Khai báo mảng có 5 phần tử
    int arr[] = {1, 2, 3, 4, 5};
    // Thử in địa chỉ của từng phần tử
    // sizeof (arr): Kích thước của mảng
    // sizeof (int): Kích thước của kiểu int
    for (int i = 0; i < sizeof(arr) / sizeof(int); i++)
    {
        printf("Dia chi cua arr[%d] = %d\n", i, &arr[i]);
    }

    printf("\n_____\n");

    // Gán con trỏ p cho phần tử đầu tiên của mảng
    int *p = &arr[0];

    // Lấy giá trị của con trỏ p
    printf("Gia tri cua con tro p = %d\n", p);
    // Lấy giá trị của địa chỉ mà p đang trỏ đến
    printf("Gia tri cua dia chi ma p dang tro den = %d\n", *p);

    // Toán tử tăng trên con trỏ
    p++; // hoặc p = p + 1;
    // Lấy giá trị của con trỏ p
    printf("Gia tri cua con tro p = %d\n", p);
    // Lấy giá trị của địa chỉ mà p đang trỏ đến
    printf("Gia tri cua dia chi ma p dang tro den = %d\n", *p);

    // Toán tử tăng trên con trỏ
    p += 2; // hoặc p = p + 2;
    // Lấy giá trị của con trỏ p
    printf("Gia tri cua con tro p = %d\n", p);
    // Lấy giá trị của địa chỉ mà p đang trỏ đến
    printf("Gia tri cua dia chi ma p dang tro den = %d\n", *p);
```

```
38
39      // Toán tử giảm trên con trỏ
40      p--; // hoặc p = p - 1;
41      // Lấy giá trị của con trỏ p
42      printf("Gia tri cua con tro p = %d\n", p);
43      // Lấy giá trị của địa chỉ mà p đang trỏ đến
44      printf("Gia tri cua dia chi ma p dang tro den = %d\n", *p);
45  }
46
```

Kết quả chạy:

```
0
1   Dia chi cua arr[0] = 6487536
2   Dia chi cua arr[1] = 6487540
3   Dia chi cua arr[2] = 6487544
4   Dia chi cua arr[3] = 6487548
5   Dia chi cua arr[4] = 6487552
6
7   ____________________________
8   Gia tri cua con tro p = 6487536
9   Gia tri cua dia chi ma p dang tro den = 1
10  Gia tri cua con tro p = 6487540
11  Gia tri cua dia chi ma p dang tro den = 2
12  Gia tri cua con tro p = 6487548
13  Gia tri cua dia chi ma p dang tro den = 4
14  Gia tri cua con tro p = 6487544
15  Gia tri cua dia chi ma p dang tro den = 3
16
```

Như bạn thấy:

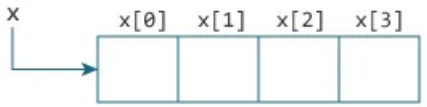
- Khi dùng toán tử tăng/ giảm trên biến con trỏ, nó sẽ nhảy sang ô nhớ liền kề chứ không phải tăng địa chỉ mà nó đang trỏ lên 1. Do con trỏ `p` là kiểu `int` nên mỗi bước tăng, giá trị của `p` tăng thêm 4 giá trị. (Lưu ý: giá trị của con trỏ là địa chỉ mà nó đang trỏ tới)
- Nếu bạn muốn tăng giá trị của địa chỉ nơi con trỏ đang trỏ tới, hãy dùng cách dưới đây:


```
0
1   #include <stdio.h>
2
3   int main()
4   {
```

```c
    // Khai báo mảng có 5 phần tử
    int arr[] = {1, 2, 3, 4, 5};

    // Gán con trỏ p cho phần tử đầu tiên của mảng
    int *p = &arr[0];
    // Lấy giá trị của địa chỉ mà p đang trỏ đến
    printf("Gia tri cua dia chi ma p dang tro den = %d\n", *p);
    // Tăng giá trị của địa chỉ mà `p` đang trỏ tới thông qua `p`.
    (*p)+= 5;
    // Lấy giá trị của địa chỉ mà p đang trỏ đến
    printf("Gia tri cua dia chi ma p dang tro den = %d\n", *p);
}

// Gia tri cua dia chi ma p dang tro den = 1
// Gia tri cua dia chi ma p dang tro den = 6

```

Mối quan hệ giữa con trỏ và mảng trong C

Tới đây chắc hẳn bạn đã hình dung được sự *liên hệ giữa con trỏ và mảng*, mình sẽ cùng các bạn đi tới các kết luận về con trỏ và mảng nhé.



Mối quan hệ giữa con trỏ và mảng trong C

Với mảng trong ảnh phía trên, ta có:

- `&x[0]` và `x` có cùng giá trị, và `x[0]` hay `*x` là tương đương nhau.
- `&x[1]` tương đương với `x+1` và `x[1]` tương đương với `*(x+1)`.
- `&x[2]` tương đương với `x+2` và `x[2]` tương đương với `*(x+2)`.
- ...
- Tóm lại, `&x[i]` tương đương với `x+i` và `x[i]` tương đương với `*(x+i)`.

Hãy thử nhập xuất mảng theo cách mới nào:

```c
// Ví dụ mối quan hệ giữa con trỏ và mảng - Lập Trình Không Khó
#include <stdio.h>

#define MAX_SIZE 100

int main()
{
    int arr[MAX_SIZE];
    int n;
    do
    {
        printf("Nhap so luong phan tu: ");
        scanf("%d", &n);
    } while (n < 1);

    // Nhập mảng
    for (int i = 0; i < n; i++)
    {
        printf("Nhap a[%d] = ", i);
        scanf("%d", (arr + i));
    }

    // Xuất mảng
    for (int i = 0; i < n; i++)
    {
        printf("\nGia tri a[%d] = %d", i, *(arr + i));
    }
}
```

Kết quả chạy:

```
Nhap so luong phan tu: 5
Nhap a[0] = 1
Nhap a[1] = 2
Nhap a[2] = 3
Nhap a[3] = 4
Nhap a[4] = 5

Gia tri a[0] = 1
Gia tri a[1] = 2
Gia tri a[2] = 3
```

```
Gia tri a[3] = 4
Gia tri a[4] = 5

```

Lưu ý: Trong hầu hết trường hợp, tên biến mảng được chuyển đổi thành 1 con trỏ. Do đó, bạn có thể sử dụng con trỏ để truy cập các phần tử của mảng. Tuy nhiên, bạn cần lưu ý **con trỏ và mảng không phải là một** nhé. Có 1 số trường hợp ngoại lệ, bạn xem chi tiết tại: When does array name doesn't decay into a pointer?

Ví dụ mối quan hệ giữa con trỏ và mảng

Bạn cũng có thể sử dụng một biến con trỏ khác để duyệt mảng, xem ví dụ bài tập tính tổng các phần tử trong mảng 1 chiều sử dụng con trỏ dưới đây:

```c

// Ví dụ mối quan hệ giữa con trỏ và mảng - Lập Trình Không Khó
#include <stdio.h>
#define MAX_SIZE 100

int main()
{
    int arr[MAX_SIZE];
    int n;
    do
    {
        printf("Nhap so luong phan tu: ");
        scanf("%d", &n);
    } while (n < 1);

    int *p = &arr[0];
    // Nhập mảng
    for (int i = 0; i < n; i++)
    {
        printf("Nhap a[%d] = ", i);
        scanf("%d", (p + i));
    }

    // Xuất mảng
    for (int i = 0; i < n; i++)
    {
        printf("\nGia tri a[%d] = %d", i, *(p + i));
    }
```

```
28
29      // Tính tổng sử dụng biến lặp là con trỏ
30      // Khởi tạo con trỏ i trỏ tới phần tử đầu tiên của mảng
31      // Ta sẽ dừng nếu giá trị của i vượt quá địa chỉ của phần tử cuối cùng
32      // i++: tăng i lên 1 đơn vị, tức là cho nó trỏ tới phần tử tiếp theo
33      int sum = 0;
34      for(int *i = &arr[0]; i <= &arr[n-1]; i++){
35          // Lấy giá trị của phần tử hiện tại bằng toán tử `*`
36          sum += *i;
37      }
38      printf("\nSum = %d", sum);
39  }
40
```

Kết quả chạy chương trình:

```
0
1   Nhap so luong phan tu: 5
2   Nhap a[0] = 1
3   Nhap a[1] = 2
4   Nhap a[2] = 3
5   Nhap a[3] = 4
6   Nhap a[4] = 5
7
8   Gia tri a[0] = 1
9   Gia tri a[1] = 2
10  Gia tri a[2] = 3
11  Gia tri a[3] = 4
12  Gia tri a[4] = 5
13  Sum = 15
14
```

Bài viết này mình đã cùng bạn đi làm rõ **mối liên hệ giữa con trỏ và mảng** trong C. Giờ đây bạn có thêm 1 cách để duyệt mảng cực ngầu. Hơn hết, bạn hiểu nhiều hơn về mối quan hệ giữa con trỏ và mảng, ứng dụng của con trỏ trong mảng. Ở các bài sau chúng ta sẽ tiếp tục khám phá về con trỏ nhé.

Tài liệu tham khảo

1. https://www.programiz.com/c-programming/c-pointers-arrays

Các bài viết trong khóa học

Bài trước: Bài 63. Con trỏ trong C

Bài sau: Bài 65. Con trỏ và hàm trong C

Nguyễn Văn Hiếu

Sáng lập cộng đồng Lập Trình Không Khó với mong muốn giúp đỡ các bạn trẻ trên con đường trở thành những lập trình viên tương lai. Tất cả những gì tôi viết ra đây chỉ đơn giản là sở thích ghi lại các kiến thức mà tôi tích lũy được.

- BẠN BÈ & ĐỐI TÁC -

Luyện Code - Tự Học Đồ Họa - Cách Học Lập Trình - VNTALKING

11