# CHƯƠNG VI: MẢNG

## MẢNG 1 CHIỀU

Mảng: là tên gọi của tập hợp các phần tử có kiểu dữ liệu là dạng Số ( int, double, float, …)

Chuỗi: là tên gọi của tập hợp các phần tử có kiểu dữ liệu là dạng chữ ( char)

### Định nghĩa

Mảng hay còn gọi là aray: Là tập hợp các phần tử có cùng kiểu dữ liệu và liền kề trong ô nhớ máy tính.

**VD:** int array[5] = {1, 3, 5, 7, 9};

Gỉa sử phần tử đầu tiên có địa chỉ là 1000.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Giá trị** | 1 | 3 | 5 | 7 | 9 |
| **Index** | 0 | 1 | 2 | 3 | 4 |
| **Địa chỉ (byte)** | 1000 | 1004 | 1008 | 1012 | 1016 |

### Khai báo mảng

Khai báo 1 mảng tĩnh 1 chiều.

**Cú pháp:**

|  |
| --- |
| <Kiểu dữ liệu> <Tên mảng>[Số phần tử]; |

**VD:**

|  |
| --- |
| int array[5];  int mang[20]; |

### Khởi tạo mảng

Khởi tạo mảng được chia thành 2 loại:

* Khởi tạo mảng tĩnh
* Khởi tạo mảng động

#### Khởi tạo mảng tĩnh

Khởi tạo mảng tĩnh bao gồm khai báo mảng và gán giá trị cho mảng (***Có thể hiện số phần tử***)

**Cú pháp:**

|  |
| --- |
| <Kiểu dữ liệu> <Tên mảng>[Số phần tử] = {Giá trị}; |

**VD:**

|  |
| --- |
| int array[5] = { 1, 3, 5, 7, 9, 11}; |

**Chú ý:** Số lượng các giá trị trong dấu ngoặc kép {} không được lớn hơn số phần tử khai báo trong dấu ngoặc vuông [].

Ngoài ra, ta có thể tách rời giữa khai báo mảng và gán giá trị thành 2 quá trình độc lập.

**VD:**

|  |
| --- |
| int array[5] ; // Khai báo mảng  // Gán giá trị cho mảng  array[0] = 1;  array[1] = 3;  array[2] = 5;  array[3] = 7;  array[4] = 9; |

**VD:** Số lượng các giá trị trong dấu ngoặc kép {} nhỏ hơn số phần tử khai báo trong dấu ngoặc vuông [].

|  |
| --- |
| int array[5] = { 1, 3, 5 };  // Chương trình tự động hiểu  // array[0] = 1;  // array[1] = 3;  // array[2] = 5;  // array[3] = 0;  // array[4] = 0; |

#### Khởi tạo mảng động

**Cú pháp:**

|  |
| --- |
| <Kiểu dữ liệu> <Tên mảng>[] = {Giá trị}; |

**VD:**

|  |
| --- |
| int array[ ] = { 1, 3, 5, 7, 9, 11, 13, 15 }; |

Nếu bạn bỏ sót kích cỡ mảng thì mảng đó đủ lớn để giữ các giá trị được khởi tạo

### Kích thước của mảng

Kích thước của mảng là tổng kích thước của các phần tử có trong mảng.

***Tổng kích thước mảng = Số lượng phần tử x Kiểu dữ kiệu của mảng.***

**VD:** Giả sử có một mảng ký tự ***char array[5];***

Mảng array có 5 phần tử trong mảng, mỗi phần tử thuộc kiểu dữ liệu char (chiếm 1 byte) => Mảng array này có tổng kích thước là 5 byte trong bộ nhớ máy tính.

**VD:** int array[6];

Mảng array có 6 phần tử trong mảng, mỗi phần tử thuộc kiểu dữ liệu int (chiếm 4 byte) => Mảng array này có tổng kích thước là 24 byte trong bộ nhớ máy tính.

**VD:** int float[7];

Mảng array có 7 phần tử trong mảng, mỗi phần tử thuộc kiểu dữ liệu float (chiếm 4 byte) => Mảng array có tổng kích thước là 28 byte trong bộ nhớ máy tính.

### Truy cập vào các phần tử của mảng

Chúng ta truy cập vào các phần tử của mảng thông qua chỉ số (index). Chỉ số của mảng luôn luôn bắt đầu từ 0 và kết thúc tại phần tử thứ N-1 ***với N phần tử trong mảng***.

Ở đây, **chỉ số** có thể hiểu là vị trí của phần tử đó trong mảng. Trong ngôn ngữ C thì **chỉ số** được tính từ đầu mảng (bên trái), bắt đầu bởi số 0 và tăng dẫn 1 đơn vị về cuối mảng.

**Giả sử:** Ta có 1 mảng ***int array [5] = { 1, 3, 5, 7, 9};*** và địa chỉ của phần tử thứ 0 là 1000

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Chỉ số trong mảng C** là một biểu thị **vị trí của phần tử trong mảng** | | | | | |
| **Giá trị** | 1 | 3 | 5 | 37 | 9 |
| **Index** | 0 | 1 | 2 | 3 | 4 |
| **Địa chỉ** | 1000 | 1004 | 1008 | 1012 | 1016 |

**Cú pháp:**

|  |
| --- |
| <Tên mảng>[chỉ số] ; |

**VD:**

|  |
| --- |
| int array[5] = { 1, 3, 5 ,7, 9};  array[0]; // Lấy giá trị của phần tử thứ 0 trong mảng array.  array[1]; // Lấy giá trị của phần tử thứ 1 trong mảng array.  array[2]; // Lấy giá trị của phần tử thứ 2 trong mảng array.  array[3]; // Lấy giá trị của phần tử thứ 3 trong mảng array.  array[4]; // Lấy giá trị của phần tử thứ 4 trong mảng array. |

### Địa chỉ các phần tử trong mảng

Để biểu thị địa chỉ các phần tử trong mảng ta cũng sử dụng ký hiệu & như đối với biến. Vì bản chất mảng là tập hợp các biến có cùng kiểu dữ liệu, nên các phần tử trong mảng cũng chính là các biến mà thôi.

Địa chỉ của mảng chính là địa chỉ phần tử đầu tiên của mảng (&mang[0]).

**VD:**

|  |
| --- |
| #include <stdio.h>  int array[5] = {1,3,5,7,9};  int main(){      for(int i = 0; i < 5; i++){          printf("Gia tri cua phan tu array[%d] la %d\n", i, array[i]);          printf("Dia chi cua phan tu array[%d] la %d\n", i, &array[i]);      }      return 0;  } |

**Kết quả:**

|  |
| --- |
| Gia tri cua phan tu array[0] la 1  Dia chi cua phan tu array[0] la 4206608  Gia tri cua phan tu array[1] la 3  Dia chi cua phan tu array[1] la 4206612  Gia tri cua phan tu array[2] la 5  Dia chi cua phan tu array[2] la 4206616  Gia tri cua phan tu array[3] la 7  Dia chi cua phan tu array[3] la 4206620  Gia tri cua phan tu array[4] la 9  Dia chi cua phan tu array[4] la 4206624 |

Chúng ta có thể thấy rằng mảng bản chất là 1 dãy các biến được xếp liên tục nhau trong bộ nhớ RAM. Ở đây mình khai báo 1 mảng 5 phần tử int (mỗi phần tử chiếm 4 bytes).

### Thao tác trên mảng

#### Xuất các phần tử của mảng

|  |
| --- |
| #include <stdio.h>  int main(){      int array[5] = {1, 3, 5, 7, 9};        for (int i = 0; i < 5; i++){          printf("array[%d] = %d \n", i, array[i]);      }      return 0;  } |

**Kết quả**

|  |
| --- |
| array[0] = 1  array[1] = 3  array[2] = 5  array[3] = 7  array[4] = 9 |

#### Nhập các phần tử của mảng

Nhập các phần tử của mảng – Thì mình sẽ có 2 trường hợp:

* Trường hợp 1 là mình có 1 cái mảng rỗng và mình nhập giá trị vào.
* Trường hợp 2 là mình đã có 1 cái mảng có sẵn và mình lại tiếp tục nhập vào cho nó (***Thay thế giá trị của mảng***).

**VD**: Trường hợp 1

|  |
| --- |
| #include<stdio.h>  int main(){      int array[5]; // Khai báo 1 mảng rỗng        // Nhập giá trị cho mảng array      printf("Nhap gia tri cua array:\n");      for (int i = 0; i < 5; i++){          printf("array[%d] = ",i);          scanf("%d", &array[i]);      }      printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");      //In các giá trị của mảng array ra màn hình      for(int i = 0; i < 5; i++){          printf("array[%d] = %d\n",i, array[i]);      }      return 0;  } |

**Kết quả**

|  |
| --- |
| Nhap gia tri cua array:  array[0] = 1  array[1] = 3  array[2] = 5  array[3] = 7  array[4] = 9  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  array[0] = 1  array[1] = 3  array[2] = 5  array[3] = 7  array[4] = 9 |

**VD**: Trường hợp 2

|  |
| --- |
| #include<stdio.h>  int main(){      int array[5] = {0, 2, 4, 6 ,8}; // Có 1 cái mảng có sẵn        // Nhập lại các giá trị cho mảng array này (Thay đổi giá trị của mảng)      printf("Thay doi cac gia tri cua array:\n");      for (int i = 0; i < 5; i++){          printf("array[%d] = ",i);          scanf("%d", &array[i]);      }      printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");      //In các giá trị của mảng array ra màn hình      for(int i = 0; i < 5; i++){          printf("array[%d] = %d\n",i, array[i]);      }      return 0;  } |

**Kết quả**

|  |
| --- |
| Thay doi cac gia tri cua array:  array[0] = 1  array[1] = 3  array[2] = 5  array[3] = 7  array[4] = 9  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  array[0] = 1  array[1] = 3  array[2] = 5  array[3] = 7  array[4] = 9 |

#### Các thao tác toán học trên các phần tử trong mảng

|  |
| --- |
| #include<stdio.h>  int main(){      int sum = 0;      int array[5] = {1, 3, 5, 7, 9};      // Lặp qua từng phần tủ của mảng.      for(int i = 0; i < 5; i++) {          sum = sum + array[i];      }      printf("Tong cac phan tu trong mang la: %d ",sum);      return 0;  } |

**Kết quả:**

|  |
| --- |
| Tong cac phan tu trong mang la: 25 |

#### Tìm kiếm trong mảng

|  |
| --- |
| #include <stdio.h>  int main(){      int array[10] = {1,2,3,4,5,6,7,8,9,10};      int Find;      printf("Nhap vao phan tu can tim: ");      scanf("%d",&Find);      for(int i = 0; i < 10; i++ ){          if(Find == array[i]){              printf("Tim thay gia tri %d tai vi tri array[%d] \n", Find, i);              break;          }      }      return 0;  } |

**Kết quả:**

|  |
| --- |
| Nhap vao phan tu can tim: 6  Tim thay gia tri 6 tai vi tri array[5] |

## MẢNG HAI CHIỀU

### Khai báo mảng hai chiều

Cú pháp 1:

|  |
| --- |
| <Kiểu dữ liệu> <Tên mảng>[Số hàng][Số cột] = {Giá trị}; |

**VD:**

|  |
| --- |
| int a[3][2] = {{3,2},{5,3},{3,1}}; |

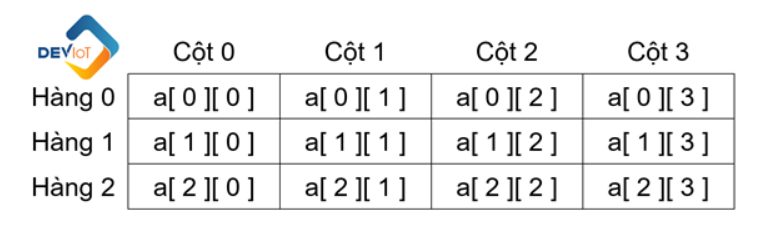
Cú pháp 2:

|  |
| --- |
| <Kiểu dữ liệu> <Tên mảng>[Số hàng][Số cột]; |

**VD:**

|  |
| --- |
| int a[3][2]; |

Một mảng 2 chiều có thể coi như là 1 bảng có x hàng và y cột. Một mảng hai chiều a chứa 3 hàng và 4 cột có thể được hiển thị như sau:



Trong đó x hàng tượng chưng cho có x mảng 1 chiều. Và y cột tượng chưng cho mỗi mảng 1 chiều có y phần tử.

### Thao tác với mảng 2 chiều

Để thao tác với mảng 2 chiều, ta có thể tự do truy cập vào từng phần tử trong mảng giống như đối với mảng 1 chiều, chú ý thứ tự hàng trước cột sau.

Chúng ta cùng tìm hiểu các thao tác với mảng 2 chiều qua ví dụ in các phần tử bên trong nó.

**VD:**

|  |
| --- |
| #include <stdio.h>    int main() {      // Khởi tạo mảng 2 chiều      int arr[3][5] = {{0, 1, 2, 3, 4}, {5, 6, 7, 8, 9}, {10, 11, 12, 13, 14}};      // Quét từng hàng      for(int i=0; i<3; i++){          // Quét từng cột          for(int j=0; j<5; j++){              // In ra từng phần tử              printf("%5d", arr[i][j]);          }          // In dấu xuống dòng          printf("\n");      }      return 0;  } |

**Kết quả:**

|  |
| --- |
| 0 1 2 3 4  5 6 7 8 9  10 11 12 13 14 |

### Địa chỉ các phần tử trong mảng

Để truy cập địa chỉ của từng phần tử trong mảng chúng ta dùng ký tự ‘&’ phía trước như 1 biến bình thường. Ví dụ địa chỉ của phần tử a[0][0] ký hiệu là &a[0][0].

**Chú ý:**

·      Địa chỉ của mảng 2 chiều trùng với địa chỉ của phần tử đầu tiên của mảng (&mang[0][0]).

·      a[0] trùng với địa chỉ của phần tử a[0][0] (tương đương &a[0][0]).

·      a[1] trùng với địa chỉ của phần tử a[1][0] (tương đương &a[1][0]).

·      a[2] trùng với địa chỉ của phần tử a[2][0] (tương đương &a[2][0]).

·      a[3] trùng với địa chỉ của phần tử a[3][0] (tương đương &a[3][0]).

**VD:**

|  |
| --- |
| #include <stdio.h>  int a[3][4] =  {      {0, 1, 2, 3},      {4, 5, 6, 7},      {8, 9, 10, 11}  };  int main() {      printf("Dia chi phan tu a[0][0] la %d\n",&a[0][0]);      printf("Dia chi phan tu a[0][0] la %d\n",a[0]);      printf("Dia chi phan tu a[0][0] la %d\n",a);      return 0;  } |

**Kết quả:**

|  |
| --- |
| Dia chi phan tu a[0][0] la 4210720  Dia chi phan tu a[0][0] la 4210720  Dia chi phan tu a[0][0] la 4210720 |

**Ví dụ 2: Nhập số phần tử mảng 2 chiều và in ra**

|  |
| --- |
| #include<stdio.h>  int main() {      int a[50][50];      int i,j,m,n;      printf("Nhap so hang n = "); scanf("%d",&n);      printf("Nhap so cot m = "); scanf("%d",&m);      printf("Nhap vao ma tran: \n");      for(i=0;i<n;i++){          for(j=0;j<m;j++){              printf("a[%d][%d] = ",i,j);              scanf("%d",&a[i][j]);          }      }      printf("ma tran vua nhap la:\n");      for(i=0;i<n;i++){          for(j=0;j<m;j++){              printf("%5d ",a[i][j]);          }          printf("\n");      }  } |

**Kết quả:**

|  |
| --- |
| **Nhap so hang n = 3**  **Nhap so cot m = 4**  **Nhap vao ma tran:**  **a[0][0] = 1**  **a[0][1] = 2**  **a[0][2] = 3**  **a[0][3] = 4**  **a[1][0] = 5**  **a[1][1] = 6**  **a[1][2] = 7**  **a[1][3] = 8**  **a[2][0] = 9**  **a[2][1] = 10**  **a[2][2] = 11**  **a[2][3] = 12**  **ma tran vua nhap la:**  **1 2 3 4**  **5 6 7 8**  **9 10 11 12** |

## TRUYỀN MẢNG VÀO HÀM

Trong ngôn ngữ C và C++ không cho phép chúng ta truyền một mảng vào hàm như là một tham số của hàm. Vậy làm sao có thể truyền mảng vào hàm được.

Câu trả lời đó là chúng ta có thể truyền vào một con trỏ trỏ tới địa chỉ của một mảng. Như vậy thật ra chúng ta đang truyền địa chỉ của mảng chứ không phải truyền toàn bộ dữ liệu của mảng vào hàm.

### Cách truyền hàm vào mảng trong C

**Cú pháp 1:** Khai báo tham số hình thức của hàm là một con trỏ.

|  |
| --- |
| KieuDuLieu TenHam(KieuDuLieu \**TenMang*) {     .     .  } |

**Cú pháp 2:** Khai báo tham số hình thức của hàm là một mảng có kích thước cụ thể.

|  |
| --- |
| KieuDuLieu TenHam(KieuDuLieu *TenMang*[KichThuoc]) {     .     .  } |

**Cú pháp 3:** Khai báo tham số hình thức của hàm là một mảng không có kích thước.

|  |
| --- |
| KieuDuLieu TenHam(KieuDuLieu *TenMang*[]) {     .     .  } |

### Ví dụ truyền hàm vào mảng trong C

**Ví dụ 1:**Viết chương trình yêu cầu người dùng nhập vào số lượng và giá trị cho các phần tử trong mảng. Sau đó viết hàm hiển thị các phần tử để in ra màn hình.

|  |
| --- |
| #include <stdio.h>  void HienThi(int *arr*[], int *n*)  {      printf("Cac phan tu cua mang: \n");      for (int i = 0; i < *n*; i++)  {          printf("   arr[%d] = %d\n",i,*arr*[i]);      }  }  int main(void) {      int n;      do{          printf("Nhap vao so luong phan tu cua mang: ");          scanf("%d", &n);          if(n <= 0) printf("Vui long nhao so luong phan tu lon hon 0 \n");      } while(n <= 0);      int a[n];      //nhập các phần tử mảng      for(int i=0; i<n; i++){          printf("Nhap vao phan tu a[%d]: ",i);          scanf("%d",&a[i]);      }      HienThi(a, n);      return 0;  } |

**Kết quả:**

|  |
| --- |
| Nhap vao so luong phan tu cua mang: 6  Nhap vao phan tu a[0]: 1  Nhap vao phan tu a[1]: 2  Nhap vao phan tu a[2]: 3  Nhap vao phan tu a[3]: 4  Nhap vao phan tu a[4]: 5  Nhap vao phan tu a[5]: 6  Cac phan tu cua mang:  arr[0] = 1  arr[1] = 2  arr[2] = 3  arr[3] = 4  arr[4] = 5  arr[5] = 6 |

**Ví dụ 2:** Viết chương trình hiển thị số nhỏ nhất và số lớn nhất trong mảng. Số lượng và giá trị của các phần tử được nhập bởi người dùng.

|  |
| --- |
| #include <stdio.h>    void HienThi(int *arr*[], int *n*)  {      printf("Cac phan tu cua mang: \n");      for (int i = 0; i < *n*; i++)  {          printf("   arr[%d] = %d\n",i,*arr*[i]);      }  }    void TimPTNhoNhat(int *arr*[], int *n*)  {      int min = *arr*[0];      for (int i = 0; i < *n*; i++)  {          if (min > *arr*[i])  {              min = *arr*[i];          }      }      printf("Phan tu nho nhat la: %d\n",min);  }  void TimPTLonNhat(int *arr*[], int *n*)  {      int max = *arr*[0];      for (int i = 0; i < *n*; i++) {          if (max < *arr*[i])  {              max = *arr*[i];          }      }      printf("Phan tu lon nhat la: %d\n",max);  }    int main(void) {      int n;      do{          printf("Nhap vao so luong phan tu cua mang: ");          scanf("%d", &n);          if(n <= 0) printf("Vui long nhap so luong phan tu lon hon 0 !!\n");      } while(n <= 0);        int a[n];      //nhập các phần tử mảng      for(int i=0; i<n; i++){          printf("Nhap vao phan tu: a[%d]: ",i);          scanf("%d",&a[i]);      }      HienThi(a, n);      printf("\n");      TimPTNhoNhat(a, n);      TimPTLonNhat(a, n);      return 0;  } |

**Kết quả:**

|  |
| --- |
| Nhap vao so luong phan tu cua mang: 7  Nhap vao phan tu: a[0]: 3  Nhap vao phan tu: a[1]: 5  Nhap vao phan tu: a[2]: 7  Nhap vao phan tu: a[3]: 8  Nhap vao phan tu: a[4]: 7  Nhap vao phan tu: a[5]: 9  Nhap vao phan tu: a[6]: 4  Cac phan tu cua mang:  arr[0] = 3  arr[1] = 5  arr[2] = 7  arr[3] = 8  arr[4] = 7  arr[5] = 9  arr[6] = 4  Phan tu nho nhat la: 3  Phan tu lon nhat la: 9 |