# CHƯƠNG VI: MẢNG

## MẢNG 1 CHIỀU

### Khai báo mảng

Có các cách sau để ta khai báo 1 mảng 1 chiều:

Cú pháp 1: Khai báo mảng không có giá trị

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

**VD:**

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

Cú pháp 2: Khai báo mảng có giá trị có số phần tử

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

**VD:**

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

Cú pháp 3: Khai báo mảng có giá trị nhưng không có số phần tử

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

**VD:**

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

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

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

Kiểu dữ liệu khi khai báo mảng sẽ quyết định kiểu dữ liệu của các phần tử trong mảng.

|  |
| --- |
| Ví dụ:  char mang1[5] nghĩa là ta có 5 phần tử của mảng có kích thước char (1 byte).  int mang1[5] nghĩa là ta có 5 phần tử của mảng có kích thước int (2 byte).  float mang1[5] nghĩa là ta có 5 phần tử của mảng có kích thước float (4 byte).  double mang1[5] nghĩa là ta có 5 phần tử của mảng có kích thước double (8 byte). |

### Truy cập vào các phần tử của mảng

Chúng ta truy cập vào các phần tử của mảng thông qua chỉ số. Chỉ số của mảng bắt đầu từ 0 và kết thúc tại phần tử thứ N-1 nếu như mảng có N phần tử.

Ở đây, **chỉ số** có thể hiểu là vị trí của phần tử đó trong mảng ban đầu. 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.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Chỉ số trong mảng C** là một biểu thị **vị trí của phần tử trong mảng** | | | | | |
| Chỉ số | 0 | 1 | 2 | 3 | 4 |
| Giá trị | 8 | 9 | 4 | 3 | 6 |

Cú pháp:

|  |
| --- |
| <Tên mảng>[chỉ số] |

**VD:**

|  |
| --- |
| Ví dụ:  int mang[5] = {0x01,0x02,0x03,0x04,0x05};  mang[0] = 0x05;        //gán giá trị 0x05 cho phần tử mang[0]  mang[1] = 0x04;        //gán giá trị 0x04 cho phần tử mang[1]  mang[2] = 0x03;        //gán giá trị 0x03 cho phần tử mang[2]  mang[3] = 0x06;        //gán giá trị 0x06 cho phần tử mang[3]  mang[4] = 0x07;        //gán giá trị 0x07 cho phần tử mang[4] |

### 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 mang[5] = {1,2,3,4,5};  int main(){      for(int i=0;i<5;i++){      printf("Gia tri cua phan tu mang[%d] la %d\n",i,mang[i]);      printf("Dia chi cua phan tu mang[%d] la %x\n",i,&mang[i]);      }      return 0;  } |

**Kết quả:**

|  |
| --- |
| Gia tri cua phan tu mang[0] la 1  Dia chi cua phan tu mang[0] la 404004  Gia tri cua phan tu mang[1] la 2  Dia chi cua phan tu mang[1] la 404008  Gia tri cua phan tu mang[2] la 3  Dia chi cua phan tu mang[2] la 40400c  Gia tri cua phan tu mang[3] la 4  Dia chi cua phan tu mang[3] la 404010  Gia tri cua phan tu mang[4] la 5  Dia chi cua phan tu mang[4] la 404014 |

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

#### Hiển thị các phần tử của mảng

|  |
| --- |
| #include <stdio.h>  int main(){      int mang[5] = {1,2,3,4,5};      for (int i = 0; i < 5; i++){          printf("mang[%d] = %d \n",i,mang[i]);      }      return 0;  } |

Kết quả

|  |
| --- |
| mang[0] = 1  mang[1] = 2  mang[2] = 3  mang[3] = 4  mang[4] = 5 |

#### Nhập các phần tử của mảng

|  |
| --- |
| #include<stdio.h>  int main(){      int mang[5];      //Nhap mang      printf("Nhap gia tri cua mang:\n");      for (int i = 0; i < 5; i++){          printf("mang[%d] = ",i);          scanf("%d", &mang[i]);      }      printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");      //In mang      for(int i = 0; i < 5; i++){          printf("mang[%d] = %d\n",i,mang[i]);      }      return 0;  } |

**Kết quả**

|  |
| --- |
| Nhap gia tri cua mang:  mang[0] = 22  mang[1] = 33  mang[2] = 44  mang[3] = 55  mang[4] = 66  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  mang[0] = 22  mang[1] = 33  mang[2] = 44  mang[3] = 55  mang[4] = 66 |

#### Các thao tác toán học trên mảng

|  |
| --- |
| #include<stdio.h>  int main(){      int mang[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};      int tong = 0;      for (int i = 0; i < 10; i++){          tong += mang[i];      }      printf("Tong cac phan tu trong mang la:%d ",tong);      return 0;  } |

**Kết quả:**

|  |
| --- |
| Tong cac phan tu trong mang la:55 |

#### Tìm kiếm trong mảng

|  |
| --- |
| #include <stdio.h>  int main(){      int a[10] = {1,2,3,4,5,6,7,8,9,10};      int phantutimkiem;      printf("Nhap vap phan tu can tim kiem: ");      scanf("%d",&phantutimkiem);      for(int i = 1; i < 11;i++ ){          if(phantutimkiem == a[i]){              printf("Tim thay gia tri %d tai vi tri a[%d] \n",phantutimkiem,i);              break;          }      }      return 0;  } |

**Kết quả:**

|  |
| --- |
| **Nhap vap phan tu can tim kiem: 5**  **Tim thay gia tri 5 tai vi tri a[4]** |

## 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}}; |

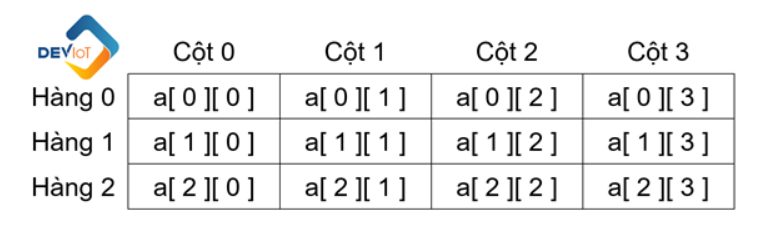
Cú pháp 2:

|  |
| --- |
| <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 |