

Bài 5.2: Mạng 2 chiều

- ✓ Mục đích sử dụng
- ✓ Cú pháp khai báo mạng
- ✓ Truy cập mảng
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Mục đích sử dụng

- ✓ Trong số các mảng nhiều chiều thì mảng 2 chiều là phổ biến nhất.
- ✓ Mục đích sử dụng: lưu trữ dữ liệu dạng bảng gồm các hàng, cột.
- ✓ Quy ước số hàng tính từ trên xuống dưới, số cột tính từ trái qua phải.
- ✓ Ứng dụng của mảng 2 chiều rất đa dạng: lưu trữ ảnh, các dữ liệu dạng bảng, bảng tính xcell, quản lý tọa độ trong game...
- ✓ Mảng 2 chiều gồm hai loại: mảng hình chữ nhật và mảng răng cưa.
- ✓ Trong mảng hình chữ nhật, tất cả các hàng có cùng kích thước hay cùng số cột.
- ✓ Trong mảng hình răng cưa, các hàng có kích thước khác nhau.

Cú pháp khai báo mảng 2 chiều

- ✓ Cú pháp khai báo mảng chữ nhật: `type[,] arrayName = new type[M, N];`
- ✓ Hoặc sử dụng var: `var arrayName = new type[M, N];`
- ✓ Cú pháp khai báo mảng răng cưa: `type[][] arrayName = new type[M][];`
- ✓ Sau đó cấp phát từng hàng thành phần i: `arrayName[i] = new type[N];`
- ✓ Trong đó:
 - ✓ Type là kiểu dữ liệu của mảng.
 - ✓ arrayName là tên mảng.
 - ✓ M, N là số hàng, số cột của mảng. M, N phải là số nguyên dương > 0 .
 - ✓ Với mảng chữ nhật, ta sử dụng cặp móc vuông có dấu phẩy bên trong.
 - ✓ Với mảng răng cưa, ta sử dụng hai móc vuông độc lập.
- ✓ Tổng số phần tử của mảng chữ nhật bằng số hàng x số cột hay $M \times N$.
- ✓ Tổng số phần tử của mảng răng cưa bằng tổng phần tử từng hàng.

Ví dụ khai báo và cấp phát

```
0 references
static void Main()
{
    // khai báo mảng chữ nhật
    int[,] matrix = new int[3, 4]; // ma trận 3 hàng 4 cột
    var friends = new string[5, 4]; // danh sách gồm 20 người bạn

    // khai báo mảng mảng của kiểu int
    var otherJaggedArr = new int[6][]; // sử dụng keyword var
    int[][] jaggedArray = new int[4][]; // cấp phát số hàng
    // cấp phát cho từng hàng
    for (int i = 0; i < jaggedArray.Length; i++)
    {
        jaggedArray[i] = new int[i + 1]; // hàng thứ i có i + 1 cột
        otherJaggedArr[i] = new int[i + 3]; // hàng thứ i có i + 3 cột
    }
}
```

Khởi tạo mảng 2 chiều

- ✓ Ta có thể khởi tạo các phần tử cho mảng 2 chiều khi biết trước giá trị các phần tử của nó.
- ✓ Với mảng chữ nhật ta có thể chỉ rõ số hàng cột hoặc bỏ qua.
- ✓ Ví dụ:

```
// khởi tạo chỉ rõ số hàng, cột
int[,] matrix = new int[3, 4] { // 3 hàng 4 cột
    { 1, 2, 3, 4 }, // mỗi hàng có 4 phần tử
    { 5, 6, 7, 8 },
    { 9, 0, 5, 2 }
};

// khởi tạo mảng không chỉ rõ số hàng, cột
var friends = new string[,]
{
    { "Nam", "Thanh", "Oanh" },
    { "Nhan", "Tung", "Khanh" },
    { "Hoang", "Quy", "Mai" }
};
```

Với mảng răng cưa

- ✓ Để khởi tạo thành công thì bắt buộc phải cấp phát số phần tử cho từng hàng.
- ✓ Ví dụ:

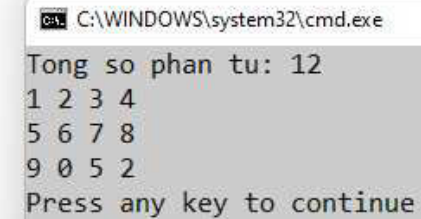
```
// khởi tạo mảng răng cưa
int[][] jaggedArr = new int[][] {
    new int[] {1, 2, 3},
    new int[] {3, 4, 5, 6},
    new int[] {7, 8, 9, 10, 12},
    new int[] {11, 12, 13, 14, 15},
};
Console.WriteLine(jaggedArr.Length); // 4
// hoặc làm theo cách sau:
int[][] jaggedArr2 = new int[3][];
jaggedArr2[0] = new int[] { 1, 2, 3 };
jaggedArr2[1] = new int[] { 1, 2, 3, 4 };
jaggedArr2[2] = new int[] { 1, 2, 3, 4, 5 };
```

Truy cập mảng 2 chiều

- ✓ Ta sử dụng tên mảng và chỉ số mảng để truy cập phần tử mảng.
- ✓ Với mảng chữ nhật, ta sử dụng cú pháp: **arrayName[rowIndex, colIndex]**
- ✓ Với mảng răng cưa, ta sử dụng cú pháp: **arrayName[rowIndex][colIndex]**
- ✓ Trong đó **rowIndex** là chỉ số hàng và **colIndex** là chỉ số cột. Cả hai chỉ số phải nằm trong đoạn $[0, N-1]$ trong đó N là kích thước tối đa của hàng/cột đang xét.
- ✓ Để lấy tổng số phần tử ta gọi thuộc tính **Length**;
- ✓ Để lấy số hàng, gọi **GetLength(0)**; Lấy số cột gọi **GetLength(1)**.
- ✓ Kết hợp với vòng lặp for lồng nhau để duyệt mảng.

Ví dụ minh họa

```
// khởi tạo chỉ rõ số hàng, cột
int[,] matrix = new int[3, 4] { // 3 hàng 4 cột
    { 1, 2, 3, 4 }, // mỗi hàng có 4 phần tử
    { 5, 6, 7, 8 },
    { 9, 0, 5, 2 }
};
// duyệt mảng
Console.WriteLine($"Tong so phan tu: {matrix.Length}");
for (int i = 0; i < matrix.GetLength(0); i++)
{
    for (int j = 0; j < matrix.GetLength(1); j++)
    {
        Console.Write(matrix[i, j] + " ");
    }
    Console.WriteLine(); // in hết hàng thì xuống dòng
}
```

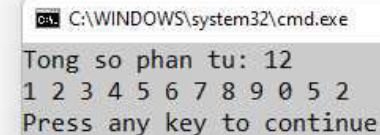


```
C:\WINDOWS\system32\cmd.exe
Tong so phan tu: 12
1 2 3 4
5 6 7 8
9 0 5 2
Press any key to continue
```


Sử dụng vòng lặp foreach

- ✓ Ta có thể sử dụng vòng lặp foreach trong trường hợp chỉ cần đọc dữ liệu ra.
- ✓ Việc sử dụng foreach sẽ làm cho mảng được duyệt từ đầu đến cuối mà không cần quan tâm chỉ số phần tử.
- ✓ Khi dùng foreach, mảng 2 chiều được coi như mảng 1 chiều.

```
// khởi tạo chỉ rõ số hàng, cột
int[,] matrix = new int[3, 4] { // 3 hàng 4 cột
    { 1, 2, 3, 4 }, // mỗi hàng có 4 phần tử
    { 5, 6, 7, 8 },
    { 9, 0, 5, 2 }
};
// duyệt mảng
Console.WriteLine($"Tong so phan tu: {matrix.Length}");
// sử dụng foreach coi mảng 2 chiều như mảng 1 chiều
foreach(var item in matrix)
{
    Console.Write(item + " ");
}
Console.WriteLine();
```



```
C:\WINDOWS\system32\cmd.exe
Tong so phan tu: 12
1 2 3 4 5 6 7 8 9 0 5 2
Press any key to continue
```



Nội dung tiếp theo

Sắp xếp và tìm kiếm trong mảng