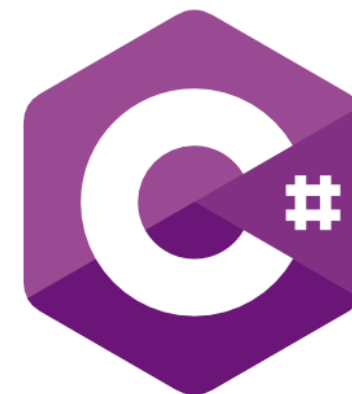


C# 5 – Mảng - Array

Giảng viên: **ThS. Lê Thiện Nhật Quang**
Email: quangln.dotnet.vn@gmail.com
Website: <http://dotnet.edu.vn>
Điện thoại: **0868.917.786**

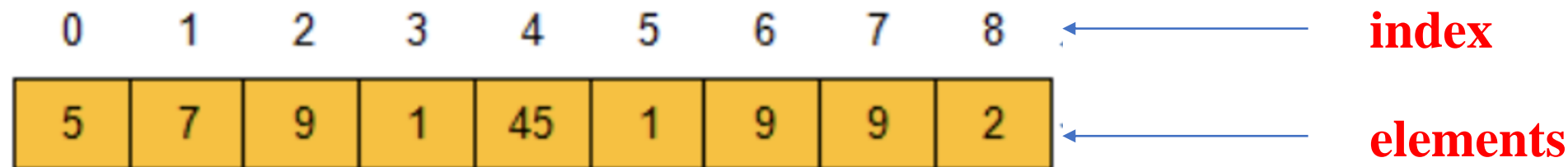


MỤC TIÊU

- Tìm hiểu về mảng
- Giải thích các loại mảng
- Giải thích lớp Array

1.1. MẢNG LÀ GÌ?

❑ Mảng là cấu trúc lưu trữ nhiều phần tử có cùng kiểu dữ liệu



❑ Để truy xuất các phần tử cần biết chỉ số (index).

❑ Trong C# chỉ số phần tử là các số nguyên không âm và bắt đầu từ 0 1 2 3...

❑ Các phần tử trong mảng dùng chung một tên.

1.1. MẢNG LÀ GÌ? (2)

❑ Các thao tác mảng

- ❖ Khai báo
- ❖ Truy xuất (đọc/ghi) phần tử
- ❖ Lấy số phần tử
- ❖ Duyệt mảng
- ❖ Tìm kiếm phần tử
- ❖ Chèn phần tử
- ❖ Sắp xếp các phần tử mảng

❑ Cú Pháp

```
// Cách 1
datatype[] arrayName; // Khai báo tên mảng và kiểu dữ liệu
arrayName = new datatype[length]; // Tạo mảng

// Cách 2
datatype[] arrayName = new datatype[length];
```

1.2. KHAI BÁO MẢNG

❑ Khai báo không khởi tạo

❖ `int[] intArray = new int[6];` // mảng số nguyên chưa biết số phần tử

❖ `string[] c = new string[5];` // mảng chứa 5 chuỗi

❑ Khai báo có khởi tạo

❖ `double[] d1 = new double[] {2, 3, 4, 5, 6};` // mảng số thực, 5 phần tử, đã được khởi tạo

❖ `double[] d2 = {2, 3, 4, 5, 6};` // mảng số thực, 5 phần tử, đã được khởi tạo

1.2. KHAI BÁO MẢNG (2)

❑ Sử dụng từ khóa Var

❖ Khởi tạo mảng có kiểu tường minh

```
var a = new int[] { 2, 6, 4, 7, 9, 3 };
```

❖ Khởi tạo mảng kiểu ngầm định

➤ Kiểu dữ liệu của mảng được hiểu ngầm định dựa vào giá trị các

phần tử

```
var a = new[] { 2, 6, 4, 7, 9, 3 }; //int[]  
var b = new[] { 2, 6, 4.5, 7, 9.5, }; //double[]  
var c = new[] { "Xin chào", "Học viên", "C#" }; //String[]
```

1.2. KHAI BÁO MẢNG (3)

❑ Sử dụng từ khóa Var

❖ Khởi tạo mảng kiểu ngầm định

- Giá trị của các phần tử phải có cùng khả năng chuyển kiểu được
- Ví dụ: khai báo sau không thể chuyển kiểu ngầm định

```
var d = new[] { 1, "Xin chào", 2, "C#" }; //Giá trị các phần tử không cùng kiểu dữ liệu
var e = new[] { 3, 2, null, 1 }; //int là kiểu giá trị, không được
var f = new[] { 3, 2, true, 1 }; //true là kiểu bool, không thể chuyển kiểu
```

1.3. TRUY XUẤT CÁC PHẦN TỬ

❑ Sử dụng chỉ số (index) để phân biệt các phần tử. Chỉ số mảng tính từ 0.

❖ `Int[] a = {4, 3, 5, 7};`

❖ `a[2] = a[1] * 4; // 3*4=12`

❖ Sau phép gán này mảng là {4, 3, 12, 7};

❑ Sử dụng thuộc tính `length` để lấy số phần tử của mảng

```
0 references
static void Main(string[] args)
{
    int[] intArray = { 3, 9, 10 };
    //Chiều dài của mảng là 3
    Console.WriteLine(intArray.Length);
}
```


1.4. DUYỆT MẢNG

□ 2 vòng lặp thường được sử dụng để duyệt mảng là for và foreach.

```
int[] a = {4, 3, 5, 9};  
for(int i=0; i<a.Length; i++){  
    Console.WriteLine(a[i]);  
}
```

for(;;)

foreach

```
int[] a = {4, 3, 5, 9};  
foreach (int x in a){  
    Console.WriteLine (x);  
}
```

1.4. DUYỆT MẢNG (2)

❑ Ví dụ sau tính tổng các số chẵn của mảng.

- ❖ Lấy từng phần tử từ mảng với foreach
- ❖ Nếu là số chẵn thì cộng vào tổng

0 references

```
static void Main(string[] args)
{
    int tong = 0;
    int[] a = { 4, 6, 8, 2, 7, 9, 5 };
    foreach (int x in a)
    {
        if (x % 2 == 0)
        {
            tong += x;
        }
    }
    Console.WriteLine(tong);
}
```

1.5. THAO TÁC MẢNG NÂNG CAO

❑ C# hỗ trợ nhiều phương thức và thuộc tính giúp lập trình nhanh hơn

```
static void Main(string[] args)
{
    string[] names = {"Jane", "Frank", "Alice", "Tom" };

    Array.Sort(names);

    foreach(string el in names)
    {
        Console.Write(el + " ");
    }

    Console.Write('\n');

    Array.Reverse(names);

    foreach(string el in names)
    {
        Console.Write(el + " ");
    }

    Console.Write('\n');
}
```

1.5. THAO TÁC MẢNG NÂNG CAO (2)

❑ C# hỗ trợ nhiều phương thức và thuộc tính giúp lập trình nhanh hơn

Phương thức	Mô tả
SetValue(value, position)	Thiết lập giá trị cho phần tử dựa vào vị trí (position)
GetValue(position)	Lấy giá trị của phần tử
IndexOf(Array, element)	Tìm kiếm một phần tử (element)
GetLength()	Cho biết số lượng phần tử trong mảng
Reverse(Array)	Đảo ngược thứ tự các phần tử
Sort(Array)	Sắp xếp các phần tử của mảng

1.6. THUẬT TOÁN SẮP XẾP

- ❑ Arrays.sort(mảng) không thể thực hiện
 - ❖ Sắp xếp giảm
 - ❖ Các kiểu không so sánh được
- ❑ Giải pháp: tự xây dựng thuật toán sắp xếp

```
int a[] = {8,2,6,2,9,1,5};  
for(int i=0; i<a.length-1; i++){  
    for(int j=i+1; j<a.length; j++){  
        if(a[i] > a[j]){  
            int temp = a[i];  
            a[i] = a[j];  
            a[j] = temp;  
        }  
    }  
}
```

Nếu thay đổi toán tử so sánh thành < thì thuật toán trở thành sắp xếp tăng dần.

2.1. MẢNG ĐA CHIỀU

- ❑ Mảng đa chiều cho phép chúng ta lưu trữ dữ liệu trên nhiều dòng
- ❑ Kích thước của mảng được xác định dựa vào số dòng và số cột

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

- ❑ Có 2 loại cơ bản là: Rectangular Array và Jagged Array

2.2. MẢNG CỐ ĐỊNH-FIXED VÀ MẢNG ĐỘNG-DYNAMIC

- Một mảng có độ dài cố định có thể lưu trữ một số lượng item được xác định trước.
- Mảng động không có kích thước được xác định trước. Kích thước của một mảng động tăng lên khi bạn thêm các item mới vào mảng. Bạn có thể khai báo một mảng có độ dài cố định hoặc động. Bạn thậm chí có thể thay đổi một mảng động thành tĩnh sau khi nó được xác định.

2.3. MẢNG THAM CHIẾU

Một biến mảng có thể được tham chiếu bởi một biến mảng khác (biến tham chiếu).

Trong khi tham chiếu, biến mảng tham chiếu đề cập đến các giá trị của biến mảng được tham chiếu.

Mã sau đây cho thấy việc sử dụng các mảng tham chiếu:

0 references

```
static void Main(string[] args)
{
    string[] classOne = { "Allan", "Chris", "Monica" };
    string[] classTwo = { "Katie", "Niel", "Mark" };
    Console.WriteLine("Students of Class I:\tStudents of Class II");
    for(int i = 0; i < 3; i++)
    {
        Console.WriteLine(classOne[i]+" \t\t\t"+ classTwo[i]);
    }
    classTwo = classOne;
    Console.WriteLine("Students of Class II after referencing Class II:");
    for (int i = 0; i < 3; i++)
    {
        Console.WriteLine(classTwo[i]+" ");
    }
    Console.WriteLine();
    classTwo[2] = "Mike";
    Console.WriteLine("Students of Class I after changing the third student in Class II:");
    for (int i = 0; i < 3; i++)
    {
        Console.WriteLine(classOne[i] + " ");
    }
    Console.ReadLine();
}
```



C:\Users\lethiennhatquang\source\repos\Demo2\Demo2\bin\Debug\Demo2.exe

Students of Class I: Students of Class II

Allan Katie

Chris Niel

Monica Mark

Students of Class II after referencing Class I:

Allan

Chris

Monica

Students of Class I after changing the third student in Class II:

Allan

Chris

Mike

2.4. RECTANGULAR ARRAY –CÙNG KÍCH THƯỚC

❑ Rectangular Array: đặc trưng là mảng 2 chiều có m dòng và n cột

❑ Cú pháp:

```
datatype[,] arrayName = new datatype[rows , columns];
```

Chiều dài = hàng * cột = $2 * 6 = 12$

2 dòng	10	5	9	7	8	2
	1	3	2	0	5	9
6 cột						

0 references

```
static void Main(string[] args)
{
    int[,] twodim = new int[,] { { 1, 2, 3 }, { 1, 2, 3 } };
    int d1 = twodim.GetLength(0);
    int d2 = twodim.GetLength(1);
    for(int i = 0; i < d1; i++)
    {
        for(int j = 0; j < d2; j++)
        {
            Console.WriteLine(twodim[i, j]);
        }
    }
}
```

1 reference

```
static void studentDetails()
{
    Console.WriteLine("Enter the number of Students: ");
    int noOfStds = Convert.ToInt32(Console.ReadLine());
    Console.WriteLine("Enter the number of Exams: ");
    int exams = Convert.ToInt32(Console.ReadLine());
    string[] stdName = new string[noOfStds];
    string[,] details = new string[noOfStds, exams];
    for (int i = 0; i < noOfStds; i++)
    {
        Console.WriteLine();
        Console.WriteLine("Enter the Student Name: ");
        stdName[i] = Convert.ToString(Console.ReadLine());
        for (int y = 0; y < exams; y++)
        {
            Console.WriteLine("Enter Score in Exam " + (y + 1) + ": ");
            details[i, y] = Convert.ToString(Console.ReadLine());
        }
    }

    Console.WriteLine();
    Console.WriteLine("Student Exam Details");
    Console.WriteLine("-----");
    Console.WriteLine();
    Console.WriteLine("Student\t\tMarks");
    Console.WriteLine("-----\t\t-----");
}
```

```
for (int i = 0; i < stdName.Length; i++)  
{  
    Console.WriteLine(stdName[i]);  
    for (int j = 0; j < exams; j++)  
    {  
        Console.WriteLine("\t\t" + details[i, j]);  
    }  
    Console.WriteLine();  
}
```

```
}
```

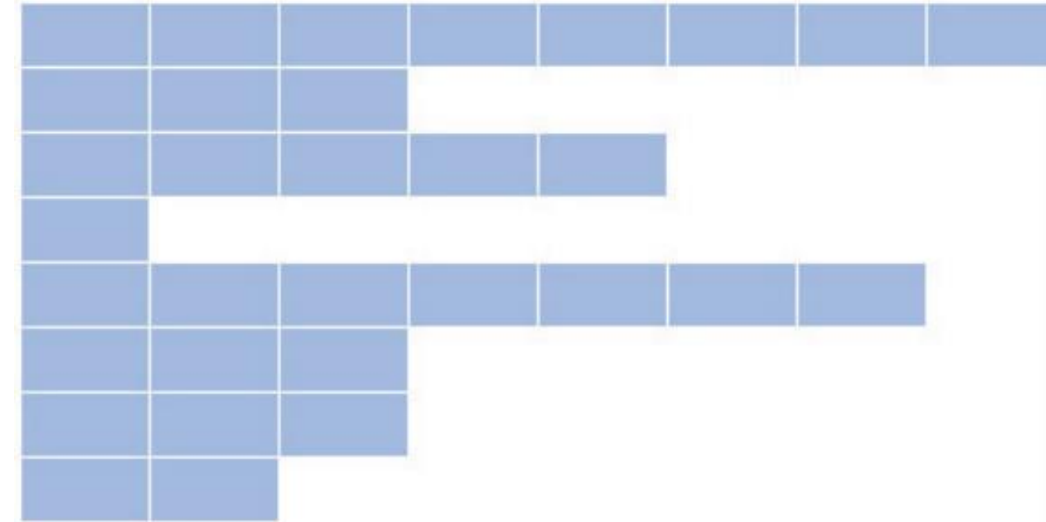
0 references

```
static void Main(string[] args)  
{  
    studentDetails();  
    Console.ReadLine();  
}
```

2.5. JAGGED ARRAY

❑ Jagged Array: tương tự Rectangular Array ngoại trừ số cột trên mỗi dòng có thể khác nhau

10	5	9	7	8	2
1	3	2	0	5	9
3	1	2			



❑ Cú pháp:

```
data_type[][] name_of_array = new data_type[rows][]
```

2.5. JAGGED ARRAY (2)

```
// Main Method
public static void Main()
{
    // Declare the Jagged Array of four elements:
    int[][] jagged_arr = new int[4][];

    // Initialize the elements
    jagged_arr[0] = new int[] {1, 2, 3, 4};
    jagged_arr[1] = new int[] {11, 34, 67};
    jagged_arr[2] = new int[] {89, 23};
    jagged_arr[3] = new int[] {0, 45, 78, 53, 99};

    // Display the array elements:
    for (int n = 0; n < jagged_arr.Length; n++) {
        // Print the row number
        System.Console.Write("Row({0}): ", n);

        for (int k = 0; k < jagged_arr[n].Length; k++) {
            // Print the elements in the row
            System.Console.Write("{0} ", jagged_arr[n][k]);
        }
        System.Console.WriteLine();
    }
}
```

Output:

```
Row(0): 1 2 3 4
Row(1): 11 34 67
Row(2): 89 23
Row(3): 0 45 78 53 99
```


0 references

```
static void Main(string[] args)
{
    /* Khai báo một jagged array gồm 5 mảng con kiểu integer */
    int[][] a = new int[][]
    {
        new int[] { 0, 0 },
        new int[] { 1, 2 },
        new int[] { 2, 4 },
        new int[] { 3, 6 },
        new int[] { 4, 8 }
    };

    /* In giá trị từng mảng con của jagged array */
    for (int row = 0; row < 5; row++)
    {
        for (int col = 0; col < 2; col++)
        {
            Console.Write("a[{0}][{1}] = {2}\t", row, col, a[row][col]);
        }
        Console.WriteLine();
    }
    Console.ReadLine();
}
```

a[0][0] = 0	a[0][1] = 0
a[1][0] = 1	a[1][1] = 2
a[2][0] = 2	a[2][1] = 4
a[3][0] = 3	a[3][1] = 6
a[4][0] = 4	a[4][1] = 8

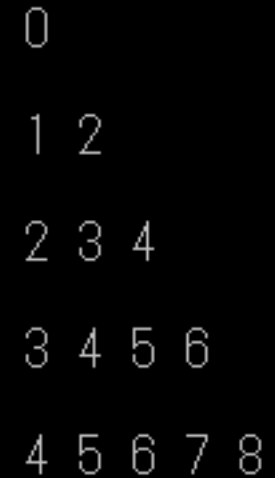

```
static void Main(string[] args)
{
    // Khai báo một jagged array A kiểu integer
    int[][] A;

    // Khởi tạo mảng A: 5 dòng, số cột chưa biết
    A = new int[5][];

    /* Khởi tạo số cột cho từng dòng:
    * row đầu tiên có 1 col
    * row thứ 2 có 2 col
    * row thứ 3 có 3 col
    * row thứ 4 có 4 col
    * row thứ 5 có 5 col
    */
    for (int row = 0; row < A.Length; row++)
    {
        A[row] = new int[row + 1];
    }

    // Nhập liệu và in mảng ra màn hình
    for (int row = 0; row < 5; row++)
    {
        for (int col = 0; col < A[row].Length; col++)
        {
            A[row][col] = row + col;
            Console.Write(A[row][col] + " ");
        }
        Console.WriteLine();
    }

    Console.ReadLine();
}
```



```
0
1 2
2 3 4
3 4 5 6
4 5 6 7 8
```

2.6. SỬ DỤNG VÒNG LẶP FOREACH

Ngoài việc sử dụng vòng lặp for, chúng ta có thể sử dụng foreach theo cú pháp bên dưới để làm việc với mảng.

Vòng lặp foreach dễ viết hơn, nó không yêu cầu bộ đếm (sử dụng thuộc tính Length) và nó dễ đọc hơn.

Cú pháp:

```
foreach (datatype variable in arrayName) {  
    // Xử lý  
}
```

0 references

```
static void Main(string[] args)  
{  
    string[] cars = { "Honda", "BMW", "Ford", "Mazda" };  
    foreach (string i in cars)  
    {  
        Console.WriteLine(i);  
    }  
    Console.ReadLine();  
}
```

3.1. CLASS ARRAY

Array cung cấp nhiều thuộc tính và phương thức để làm việc với mảng. Bảng sau đây trình bày một số phương thức thường dùng của lớp Array.

Lớp Array trong C# là mẹ của tất cả các mảng và cung cấp chức năng để tạo, thao tác, tìm kiếm và sắp xếp mảng trong .NET Framework.

Lớp Array, được định nghĩa trong namespace System, là lớp cơ sở cho mảng trong C#. Lớp Array là một lớp cơ sở trừu tượng có nghĩa là chúng ta không thể tạo instance của lớp Array.

3.2. THUỘC TÍNH VÀ PHƯƠNG THỨC CỦA ARRAY

IsFixedSize	Trả về giá trị boolean, một giá trị cho biết một mảng có kích thước cố định hay không.
IsReadOnly	Trả về giá trị boolean, một giá trị cho biết một mảng có ở chế độ chỉ đọc hay không.
IsSynchronized	Trả về giá trị boolean, một giá trị cho biết một mảng có hàm xử lý đa luồng cùng lúc. Giá trị mặc định là false
LongLength	Trả về một số nguyên 64 bit đại diện cho tổng số mục trong tất cả các kích thước của một mảng.
Length	Trả về một số nguyên 32 bit đại diện cho tổng số mục trong tất cả các kích thước của một mảng.
Rank	Trả về số thứ tự nguyên của một mảng.
SyncRoot	Trả về một đối tượng sử dụng để đồng bộ hóa truy cập mảng

Thuộc tính của Array

3.2. THUỘC TÍNH VÀ PHƯƠNG THỨC CỦA ARRAY (2)

Phương thức	Mô tả
SetValue(value, position)	Thiết lập giá trị cho phần tử dựa vào vị trí (position)
GetValue(position)	Lấy giá trị của phần tử
IndexOf(Array, element)	Tìm kiếm một phần tử (element)
GetLength()	Cho biết số lượng phần tử trong mảng
Reverse(Array)	Đảo ngược thứ tự các phần tử
Sort(Array)	Sắp xếp các phần tử của mảng
Clear	Xóa tất cả các phần tử trong mảng và thiết lập kích cỡ mảng là 0, về false hoặc về null
CopyTo	Sao chép tất cả các phần tử hiện tại của mảng 1 chiều đến mảng 1 chiều khác từ vị trí index cụ thể
GetLowerBound	Trả về giới hạn dưới cho mảng tương ứng
GetUpperBound	Trả về giới hạn trên cho mảng tương ứng
Initialize	Khởi tạo mỗi phần tử của mảng bằng cách gọi hàm dựng mặc định của lớp Array

Phương thức của Array

3.3. SỬ DỤNG LỚP ARRAY

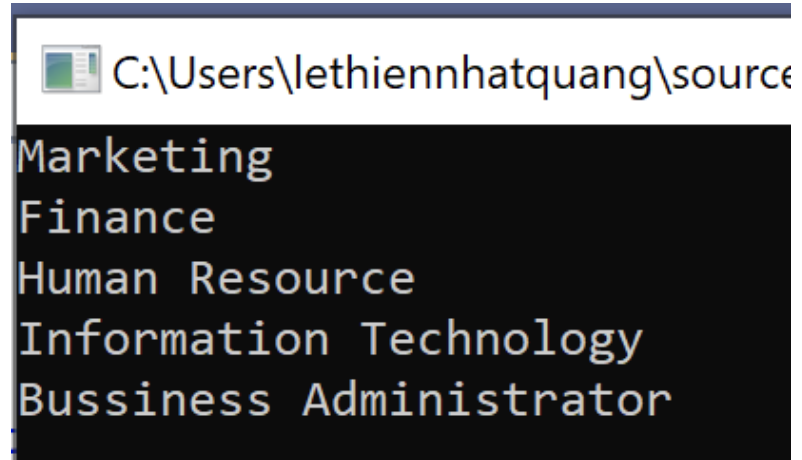
Lớp Array cung cấp phương thức CreateInstance để xây dựng một mảng. Phương thức CreateInstance nhận tham số đầu tiên là kiểu item và tham số thứ hai và thứ ba là thứ nguyên và phạm vi của chúng. Khi một mảng được tạo, chúng ta sử dụng phương thức SetValue để thêm các item vào mảng.

```
Array stringArray = Array.CreateInstance(typeof(String), 3);
stringArray.SetValue("Mahesh Chand", 0);
stringArray.SetValue("Raj Kumar", 1);
stringArray.SetValue("Neel Beniwal", 2);
```

```
Array intArray3D = Array.CreateInstance(typeof(Int32), 2, 3, 4);
for (int i = intArray3D.GetLowerBound(0); i <= intArray3D.GetUpperBound(0); i++)
    for (int j = intArray3D.GetLowerBound(1); j <= intArray3D.GetUpperBound(1); j++)
        for (int k = intArray3D.GetLowerBound(2); k <= intArray3D.GetUpperBound(2); k++)
        {
            intArray3D.SetValue((i * 100) + (j * 10) + k, i, j, k);
        }
foreach(int ival in intArray3D) {
    Console.WriteLine(ival);
}
```

0 references

```
static void Main(string[] args)
{
    Array objArray = Array.CreateInstance(typeof(string), 5);
    objArray.SetValue("Marketing", 0);
    objArray.SetValue("Finance", 1);
    objArray.SetValue("Human Resource", 2);
    objArray.SetValue("Information Technology", 3);
    objArray.SetValue("Bussiness Administrator", 4);
    for (int i=0; i<=objArray.GetUpperBound(0); i++)
    {
        Console.WriteLine(objArray.GetValue(i));
    }
    Console.ReadLine();
}
```



C:\Users\lethiennhatquang\source

Marketing
Finance
Human Resource
Information Technology
Bussiness Administrator

Để thao tác mảng, lớp Array sử dụng bốn giao diện sau:

- Giao diện **Iconeable** thuộc không gian tên System và chứa phương thức Clone() cho phép tạo bản sao chính xác của đối tượng hiện tại của lớp
- Giao diện **Icollection** thuộc không gian tên System.Collections và chứa các thuộc tính cho phép đếm số lượng phần tử, kiểm tra xem các phần tử có được đồng bộ hóa hay không, sau đó đồng bộ hóa các phần tử trong collection
- Giao diện **IList** thuộc không gian tên System.Collections và cho phép tùy chỉnh việc định nghĩa các phần tử trong mảng. Giao diện này định nghĩa 3 thuộc tính: IsFixedSize, IsReadOnly, và Item.
- Giao diện **IEnumerable** thuộc không gian tên System.Collections. Giao diện này trả về một enumerator có thể sử dụng với vòng lặp foreach để lặp lại qua một tập hợp các phần tử chẳng hạn như một mảng

3.4. RANK

Rank là thuộc tính read-only chỉ định số chiều của mảng
Ví dụ: Một mảng 3 chiều có 3 rank

References

```
static void Main(string[] args)
{
    int[] arr = new int[] { 15, 20, 5, 25, 10 };

    Console.WriteLine("Rank = {0}, Length = {1}", arr.Rank, arr.Length);
    Console.ReadLine();
}
```



Select C:\Users\lethiennhatquang\

```
Rank = 1, Length = 5
```

BÀI TẬP

Bài 1: Viết chương trình nhập mảng số nguyên từ bàn phím theo 2 cách: sử dụng mảng 1 chiều thông thường và sử dụng ArrayList.

- ✓ Sắp xếp và xuất mảng vừa nhập ra màn hình.
- ✓ Xuất phần tử có giá trị nhỏ nhất ra màn hình
- ✓ Tính và xuất ra màn hình trung bình cộng các phần tử chia hết cho 3

BÀI TẬP

Bài 2: Viết chương trình nhập 2 mảng họ tên và điểm của sinh viên

✓ Xuất 2 mảng đã nhập, mỗi sinh viên có thêm học lực

- o Yếu: điểm < 5

- o Trung bình: $5 \leq \text{điểm} < 6.5$

- o Khá: $6.5 \leq \text{điểm} < 7.5$

- o Giỏi: $7.5 \leq \text{điểm} < 9$

- o Xuất sắc: điểm ≥ 9

✓ Sắp xếp danh sách sinh viên đã nhập tăng dần theo điểm

HƯỚNG DẪN

✓ Sử dụng lệnh if để xét học lực sau đó xuất thông tin từng sinh viên

- o Họ tên:

- o Điểm:

- o Học lực:

✓ Bài này bạn không thể sử dụng `Arrays.sort()` để sắp xếp được mà phải sử dụng đến thuật toán tùy biến (tham khảo slide bài giảng)