Bài 5.1: Mảng 1 chiều

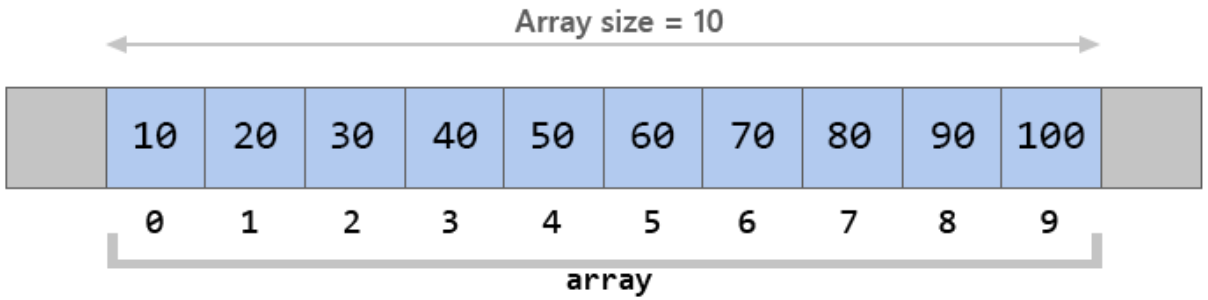
- ✓ Định nghĩa và mục đích sử dụng
- ✓ Cú pháp khai báo và khởi tạo
- ✓ Truy cập mảng
- ✓ Vòng lặp foreach
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Định nghĩa và mục đích sử dụng

- ✓Mảng là một tập hợp của các phần tử cùng kiểu được cấp phát các vùng nhớ liên tiếp nhau và truy cập qua chỉ số phần tử.
- ✓Cụ thể hơn, bộ nhớ được cấp cho mảng là một loạt vùng nhớ liên tiếp nhau cùng kích thước được chỉ định bởi kiểu dữ liệu của các phần tử trong mảng.
- ✓Mảng được sử dụng để lưu trữ tập hợp các phần tử: danh sách người yêu cũ, danh sách các môn học, danh sách các món đồ công nghệ bạn sở hữu,...
- ✓Mỗi thành viên trong tập hợp cấu thành nên mảng gọi là phần tử của mảng.
- ✓Vị trí của phần tử mảng gọi là chỉ số mảng, bắt đầu từ 0 và kết thúc ở n-1.
- ✓Ta có thể khai báo mảng kiểu bất kì khác void trong C#.
- ✓Tổng số phần tử tối đa có thể chứa trong mảng là cố định không thể thay đổi, gọi là kích thước mảng. Được tính bằng số phần tử của mảng.
- ✓Để lấy kích thước mảng ta sử dụng cú pháp: arrayName.Length

Ví dụ về mảng

✓Ví dụ sau là mảng các số nguyên kích thước 10 phần tử.



Cú pháp khai báo và khởi tạo

- ✓Cú pháp khai báo và cấp phát mảng: dataType[] arrayName = new dataType[N];
- ✓Hoặc sử dụng kiểu ngầm định var: var arrayName = new dataType[N];
- ✓Trong đó:
  - ✓ dataType: là kiểu dữ liệu của mảng, có thể là int, float, string, …
  - ✓ arrayName: tên mảng, thường đại diện cho cả tập hợp nên là danh từ số nhiều.
  - ✓ Keyword new là bắt buộc.
  - ✓ N là số phần tử tối đa hay kích thước mảng, đặt trong cặp [] sau kiểu dữ liệu. N phải là số nguyên > 0.
  - ✓ Khi sử dụng cách 1, phải có cặp [] ngay sau tên kiểu dữ liệu.
  - ✓ Khi sử dụng keyword var, không có sự xuất hiện của cặp [] ngay sau var.
- ✓Tất cả các biến mảng phải được cấp phát với keyword new hoặc được khởi tạo trước khi sử dụng.
- ✓Khi cấp phát, các phần tử trong mảng sẽ nhận giá trị mặc định của kiểu mảng: số là 0, bool là false, các kí tự là '\0', các kiểu tham chiếu là null.

Ví dụ khai báo và cấp phát mảng

✓Ví dụ sau khai báo và cấp phát mảng với 2 cách khác nhau:

```
0 references
static void Main()
{
    int[] numbers = new int[100]; // mảng int chứa tối đa 100 số nguyên
    float[] grades = new float[50]; // mảng float chứa tối đa 50 đầu điểm tb
    string[] exGirlFriends = new string[500]; // mảng string tối đa 500 nyc
    // khai báo mảng với var:
    var friends = new string[200]; // mảng string tối đa tên 200 bạn bè
    var students = new string[100]; // mảng string tối đa tên 100 sinh viên
}
```

Khởi tạo mảng

- ✓Khi tạo biến mảng, nếu biết trước giá trị các phần tử mảng hoặc muốn tạo giá trị ban đầu cho chúng, ta có thể khởi tạo luôn giá trị cho mảng.
- ✓Các giá trị này phải cùng kiểu với kiểu của mảng.
- ✓Cách 1: dataType[] arrayName = {element1, element2, …, element k};
- ✓Cách 2: dataType[] arrayName = new dataType[] {element1, element2, …, element k};
- ✓Cách 3: var arrayName = new dataType[] {element1, element2, …, element k};
- ✓Lúc này số phần tử liệt kê trong cặp {} phân tách bằng dấu phẩy sẽ là các phần tử của mảng.
- ✓Kích thước mảng được xác định tự động qua số lượng phần tử này. Ta không cần chỉ định số phần tử trong [] bên phải dấu =.

Ví dụ khai báo và khởi tạo mảng

✓Ví dụ sau khai báo và khởi tạo mảng với 2 cách khác nhau:

```
// khởi tạo mảng int
int[] ages = { 1, 5, 25, 34, 27, 31, 50, 69, 71 };
// khởi tạo mảng string
string[] programmingLanguages = new string[] { "C", "C++", "Java", "Python", "C#", "JavaScript" };
// khởi tạo mảng double
var interestRate = new double[] { 1.25, 3.45, 4.75, 5.25, 6.5, 7.25 };
// khởi tạo sau bị lỗi do số lượng phần tử thực tế và kích thước chỉ định khác nhau
int[] months = new int[12] { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11};
```

Truy cập mảng

- ✓Ta sử dụng tên mảng và chỉ số phần tử của mảng để truy cập và gán giá trị cho các phần tử trong mảng.
- ✓Chỉ số luôn nằm trong đoạn [0, n-1] với n là số phần tử của mảng.
- ✓Khi duyệt mảng ta thường sử dụng kết hợp với vòng lặp for vì có biến kiểm soát lặp.
- ✓Ví dụ:

```
var friends = new string[200]; // mảng string tối đa tên 200 bạn bè
// khởi tạo giá trị cụ thể cho mảng
friends[0] = "Nam";
friends[1] = "Mai";
friends[199] = "Thanh"; // phần tử cuối của mảng friends
friends[200] = "Loan"; // lỗi vì 200 nằm ngoài biên mảng
//lỗi: System.IndexOutOfRangeException: Index was outside the bounds of the array.

// vòng lặp hiển thị các phần tử của mảng friends
for (int i = 0; i < friends.Length; i++)
{
    Console.WriteLine(friends[i]);
}
```

Vòng lặp foreach

- ✓Để làm việc với tập hợp, ví dụ như mảng, list, collections ta có thể sử dụng vòng lặp for thường hoặc foreach.
- ✓Dùng for thường nếu muốn thao tác với vị trí của phần tử trong tập hợp. Hoặc muốn kiểm soát việc duyệt một phần của mảng.
- ✓Dùng foreach nếu không quan tâm đến chỉ số của phần tử mảng.
- ✓Theo mặc định foreach duyệt mảng ở chế độ chỉ đọc, bắt đầu từ phần tử đầu tiên trong mảng đến hết thì dừng lại.
- ✓Cú pháp: foreach(var item in collection) { statements; }
- ✓Trong đó:
  - ✓ Collection là tên mảng hoặc tập hợp cần duyệt.
  - ✓ Có thể thay var bởi kiểu của mảng
  - ✓ Item là tên đại diện cho 1 phần tử trong tập hợp, có thể thay đổi sang tên khác.
  - ✓ Statements là các lệnh ở chế độ chỉ đọc. Không thay đổi giá trị các phần tử trong foreach.

Ví dụ sử dụng foreach

✓Ví dụ duyệt mảng bằng foreach:

```
var friends = new string[200]; // mảng string tối đa tên 200 bạn bè
// khởi tạo giá trị cụ thể cho mảng
friends[0] = "Nam";
friends[1] = "Mai";
friends[20] = "Loan";
friends[100] = "Hai";
friends[199] = "Thanh"; // phần tử cuối của mảng friends

// duyệt mảng sử dụng foreach
foreach (var item in friends)
{
    Console.WriteLine(item);
}

// cú pháp sau sẽ bị lỗi
foreach (var item in friends)
{
    item = "New Info"; // error!
    Console.WriteLine(item);
}
```

Nội dung tiếp theo

Mảng nhiều chiều trong C#