CHƯƠNG II:

NỀN TẢNG NGÔN NGỮ LẬP TRÌNH C#

1

5. CÁC KIỂU DỮ LIỆU CÓ CẤU TRÚC

- MẢNG
- COLLECTION
- KIỂU CHUỖI
- FILE VĂN BẢN

2

5.1. MẢNG

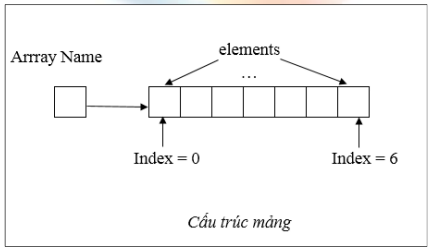
Giới thiệu:

- Mảng là một tập hợp có thứ tự các biến có cùng kiểu dữ liệu.
- Các biến trong mảng được gọi là các phần tử của mảng
- Các phần tử của mảng được truy cập bằng cách sử dụng tên mảng và chỉ số (index) của mảng
- Mảng là một kiểu dữ liệu tham chiếu

3

5.1. MẢNG

Giới thiệu:



Cấu trúc mảng

4

5.1. MẢNG

Khai báo mảng:

Mảng phải được khai báo trước khi sử dụng

- ***Cú pháp:***

  kiểu_dữ_liệu [ ] tên_mảng;

  kiểu_dữ_liệu: xác định kiểu dữ liệu của các phần tử trong mảng

  [ ]: xác định số chiều (rank) của mảng

  tên_mảng: xác định tên của mảng

- ***Ví dụ***: int[ ] Score;

5

5.1. MẢNG

Khởi tạo và gán giá trị cho mảng

```
//khai báo mảng
int[] Score;
//khởi tạo mảng có 10 phần tử
Score = new int[10];
//hoặc khai báo và khởi tạo trong 1 câu lệnh
int[] Score = new int[10];
//gán giá trị 4 cho phần tử đầu tiên của mảng
Score [0] = 4;
```

6

Khởi tạo và gán giá trị cho mảng

```
//gán giá trị cho mảng khi khai báo
int[] Score = {5, 10, 15, 20, 25, 30, 35, 40, 45, 50 };
//hoặc
int[] Score = new int[10] { 5, 10, 15, 20, 25, 30,
                            35,40, 45, 50 };
//hoặc
int[] Score = new int[] { 5, 10, 15, 20, 25, 30, 35,
                          40, 45, 50 };
```


Sao chép mảng

- Khi copy mảng, cả biến nguồn và đích đều tham chiếu đến cùng một thể hiện mảng trong bộ nhớ
- Ví dụ:

```
int[] mangNguon= new int[] { 1, 3, 5, 7};
int[] mangDich= mangNguon;
```


Ví dụ:

Viết chương trình cho phép người dùng nhập n số nguyên vào từ bàn phím. Hiển thị dãy số đã nhập và tổng các số đó.

Các phương thức:

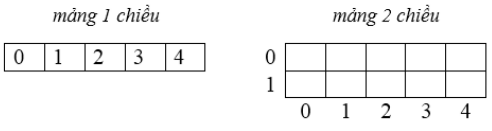
- *Nhập các phần tử của mảng(số phần tử của mảng) trả về mảng 1 chiều …*
- *Hiển thị mảng (mảng 1 chiều)*
- *Tính tổng các phần tử của mảng (mảng 1 chiều) trả về tổng tính được*

Sử dụng foreach

- Cho phép duyệt qua tất cả các phần tử của mảng một cách đơn giản, rõ ràng
- ***Cú pháp***

```
foreach (kiểu_dữ_liệu  tên_phần_tử  in
tên_mảng)
{
    // các lệnh
}
```


Mảng 2 chiều



Cấu trúc mảng 1 chiều và mảng 2 chiều

Mảng 2 chiều

- ***Cú pháp khai báo:***

  kiểu_dữ_liệu**[,]** tên_mảng **;**

- ***Tham chiếu đến 1 phần tử:***

  tên_mảng**[**chỉ_số_dòng**,** chỉ_số_cột**]**

5.1. MẢNG

Khởi tạo mảng 2 chiều

```
//Khai báo mảng 2 chiều
int[,] numbers;
//khởi tạo mảng 2 chiều 3 dòng 2 cột
numbers = new int[3, 2];
//khai báo và khởi tạo mảng 2 chiều
int[,] numbers4 = new int[3, 2];
//gán giá trị cho phần tử ở dòng 1 cột 1 bằng 5
numbers[0, 0] = 5;
```

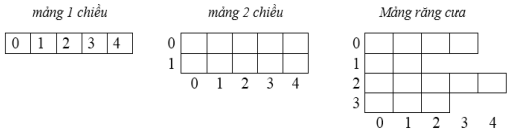

5.1. MẢNG

Khởi tạo mảng 2 chiều

```
//khai báo, khởi tạo và gán giá trị cho các phần tử
int[,] numbers1 = new int[3, 2] { {1, 2}, {3,4}, {5,6} };
//hoặc
int[,] numbers2 = new int[ , ] { {1,2},{ 3, 4 },{ 5, 6 } };
//hoặc
int[,] numbers3 = { { 1, 2 }, { 3, 4 }, { 5, 6 } };
```


5.1. MẢNG

Mảng răng cưa (jagged array)



Cấu trúc mảng 1 chiều, 2 chiều và mảng răng cưa

5.1. MẢNG

Mảng răng cưa (jagged array)

- ***Cú pháp khai báo và khởi tạo:***

  kiểu_dữ_liệu**[ ] [ ]** tên_mảng **=**
  new kiểu_dữ_liệu**[**số_hàng**] [ ] ;**

- ***Tham chiếu đến 1 phần tử:***

  tên_mảng**[**chỉ_số_dòng**][**chỉ_số_cột**]**

5.1. MẢNG

Mảng răng cưa (jagged array)

```
//khai báo và khởi tạo mảng răng cưa với ba hàng có độ dài khác nhau
int[][] numbers =  new int[3][];
numbers[0] = new int[3]; // số cột hàng thứ nhất
numbers[1] = new int[4]; // số cột hàng thứ hai
numbers[2] = new int[2]; // số cột hàng thứ ba
//gán giá trị 4 cho phần tử đầu tiên của mảng
numbers[0][0] = 4;
//khai báo và khởi tạo trong 1 câu lệnh
int[][] numbers1 = { new int[] { 1, 2, 3 },
                     new int[] { 4, 5, 6, 7 },
                     new int[] { 8, 9 } };
```


5.1. MẢNG

Array class

- Lớp Array là lớp cơ sở cho tất cả các mảng trong C#.
- Được định nghĩa trong **System** namespace
- Cung cấp các thuộc tính và phương thức để làm việc với mảng

5.1. MẢNG

Array class

Thuộc tính	Ý nghĩa
Length	Trả lại tổng số phần tử của mảng
Rank	Trả lại số chiều của mảng

Phương thức	Ý nghĩa
Sort( )	Sắp xếp mảng
Clear( )	Xóa tất cả các phần tử của mảng
Reverse	Đảo ngược thứ tự các phần tử trong mảng 1 chiều

5.1. MẢNG

Array class

Phương thức	Ý nghĩa
GetLength( )	Trả lại số phần tử trong mảng
GetValue( )	Trả lại giá trị của phần tử chỉ định trong mảng
IndexOf( )	Trả lại chỉ số của giá trị lần đầu tiên xuất hiện trong mảng 1 chiều hoặc trong một phần của mảng

5.2. COLLECTIONS

Hạn chế của mảng

- ***Thay đổi kích thước*** của mảng: tạo một mảng mới → copy các phần tử sang mảng mới → tham chiếu lại mảng
- ***Xóa một phần tử*** của mảng: xóa phần tử → di chuyển các phần tử còn lại lên
- ***Thêm một phần tử*** vào giữa mảng: phải di chuyển các phần tử xuống cuối mảng để có một chỗ trống (có thể mất phần tử cuối cùng của mảng) → thêm phần tử

5.2. COLLECTIONS

Mảng	Collection
Giống nhau: Cả hai đều có thể lưu trữ nhiều phần tử, có thể là kiểu giá trị hoặc kiểu tham chiếu	

5.2. COLLECTIONS

Các lớp collection thông dụng

- Collection không định kiểu (untyped collection)
  - Nằm trong namespace **System.Collecions**
  - Có thể lưu trữ một kiểu đối tượng bất kỳ trong collection
- Collection định kiểu (typed collection)
  - Nằm trong namespace **System.Collecions.Generic**
  - Chỉ có thể chứa các phần tử cùng kiểu dữ liệu

5.2. COLLECTIONS

Các lớp collection thông dụng

.NET 1.x	Từ .NET 2.0	Mô tả
ArrayList	List<T>	Sử dụng chỉ số để truy cập đến các phần tử.
SortedList	SortedList<K,V>	Sử dụng khóa để truy cập tới giá trị, giá trị có thể là kiểu đối tượng bất kỳ
Queue	Queue<T>	Hàng đợi
Stack	Stack<T>	Ngăn xếp

5.2. COLLECTIONS

ArrayList (danh sách mảng)

- Các phần tử của ArrayList được lưu dưới dạng kiểu object (quá trình boxing)
- Khi truy xuất tới các phần tử của mảng phải ép kiểu đối tượng sang kiểu dữ liệu tương ứng (quá trình unboxing)

25

5.2. COLLECTIONS

ArrayList (danh sách mảng)

- **Cú pháp khai báo và khởi tạo**

```
ArrayList tên_danh_sách_mảng;
tên_danh_sách_mảng = new ArrayList();
```

26

5.2. COLLECTIONS

ArrayList (danh sách mảng)

Chỉ mục	Mô tả
[index]	Lấy hoặc thiết lập phần tử có chỉ số xác định. Chỉ số của phần tử đầu tiên = 0

Thuộc tính	Mô tả
Count	Trả lại số phần tử có trong danh sách

27

5.2. COLLECTIONS

ArrayList

Phương thức	Sử dụng
Add( )	Thêm một phần tử vào cuối ArrayList
Insert( )	Thêm một phần tử vào vị trí xác định
Remove( )	Xóa phần tử lần đầu tiên xuất hiện trong ArrayList
RemoveAt( )	Xóa phần tử có chỉ số xác định
Clear( )	Xóa tất cả các phần tử khỏi ArrayList
Contain( )	Trả về giá trị Boolean cho biết danh sách có chứa đối tượng cần tìm không

28

5.2. COLLECTIONS

List<T> - Danh sách các phần tử có kiểu T

- **Cú pháp khai báo**

```
List< kiểu_dữ_liệu> tên_danh_sách;
```

- **Khởi tạo danh sách**

```
tên_danh_sách = new List< kiểu_dữ_liệu>( );
```

- Các phương thức,thuộc tính tương tự như ArrayList

29

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

- Tạo chuỗi sử dụng lớp **String**
- Sử dụng các thuộc tính và phương thức của lớp String để làm việc với chuỗi

Hoặc

- Tạo chuỗi sử dụng lớp **String Builder**
- Sử dụng các thuộc tính và phương thức của lớp String Builder để làm việc với chuỗi

30

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Lớp String

- **Khai báo và khởi tạo chuỗi**

```
//Khai báo và khởi tạo chuỗi
string hoTen = "Tran Van A";
//Khai báo và khởi tạo chuỗi rỗng
string chon = "";
string tiepTuc = null;
string loai = string.Empty;
```


4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Lớp String

- **Nối chuỗi và ghép chuỗi**

```
//nối chuỗi
string ho = "Tran";
string ten = " A";
string hoVaTen = ho + " " + ten;//kêt quả là Tran A
//ghép chuỗi
string thuDo = "Ha";
thuDo += " Noi"; //kết quả là Ha Noi
```


4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Lớp String

- **Đưa ký tự đặc biệt vào chuỗi**

Chuỗi điều khiển	Mô tả
\n	Dòng mới
\t	Tab
\\	Dấu gạch chéo
\”	Dấu nháy kép

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Lớp String

- **Chuỗi nguyên mẫu:** Sử dụng ký tự @ ngay trước dấu “ chứa nội dung chuỗi

```
string hoTen= @"Nguyen Van
Hoang";
```

➔ chuỗi là : Nguyen Van
Hoang

```
string hoTen= @"d:\demoC#\diem.txt";
```

➔ chuỗi là : d:\demoC#\diem.txt

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Lớp String

Chỉ mục	Mô tả
[index]	Trả về ký tự tại một vị trí xác định

Thuộc tính	Mô tả
Length	Trả về số ký tự trong chuỗi

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Lớp String

Phương thức	Mô tả
ToLower()	Trả về một chuỗi với các ký tự viết thường
ToUpper()	Trả về một chuỗi với các ký tự viết hoa

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Lớp String

Phương thức	Mô tả
Split(split character)	Trả về một mảng chuỗi, trong đó mỗi phần tử của mảng là một chuỗi con, được phân tích bởi một hoặc nhiều ký tự xác định
Substring(startIndex [,legth])	Trả lại một chuỗi con bắt đầu từ vị trí chỉ định và có độ dài xác định)

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Phương thức Format của String

Cú pháp:

String.Format(chuỗi_ký_tự, giá_trị_1[,giá_trị_2]. . .)

- chuỗi_ký_tự: chứa đặc tả định dạng cho một hoặc nhiều giá trị cần định dạng
- giá_trị_n: giá trị cần định dạng

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Cú pháp chuỗi đặc tả định dạng

{N[,M] [:Chuỗi_định_dạng]}

- N: Số nguyên biểu thị cho giá trị cần định dạng
- M: số nguyên biểu thị cho độ rộng của giá trị định dạng. M âm → chuỗi giá trị được căn trái. M dương→ chuỗi giá trị được căn phải
- Chuỗi_định_dạng: chuỗi các mã định dạng

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Các mã định dạng số chuẩn

Mã	Mô tả
C hoặc c	Định dạng số dưới dạng tiền tệ với số chữ số phần thập phân xác định
F hoặc f	Định dạng số dưới dạng số thập phân với số chữ số phần thập phân xác định
P hoặc p	Định dạng một số dưới dạng phần trăm với số chữ số phần thập phân xác định
N hoặc n	Định dạng một số với các dấu phân tích phần nghìn và số chữ số thập phân xác định

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Mã định dạng DateTime chuẩn

Mã	Mô tả
d	Định dạng ngày tháng ngắn
D	Định dạng ngày tháng dài
t	Định dạng thời gian ngắn
T	Định dạng thời gian dài
. . .	

4.3. LÀM VIỆC VỚI DỮ LIỆU KIỂU CHUỖI

Mã định dạng DateTime tùy chỉnh

Mã	Mô tả
d | dd	Ngày của tháng không có | có số 0 đứng đầu
ddd | dddd	Tên ngày viết tắt | viết đầy đủ
M | MM	Tháng không có | có số 0 đứng đầu
MMM |MMMM	Tên tháng viết tắt |viết đầy đủ
yy | yyyy	Số năm với hai | bốn chữ số

4.4. THAO TÁC VỚI FILE VĂN BẢN

- Namespace **System.IO** cung cấp các class để làm việc với file, quản lý thư mục . . .
- **File văn bản**: file chứa các ký tự (chuỗi) văn bản. Các trường (field) được tách bằng các ký tự đặc biệt như **tab** hoặc **|**, các bản ghi (record) được phân biệt bằng ký tự xuống dòng mới.
- **Luồng dữ liệu** (stream) là dòng dữ liệu đi từ nơi này đến nơi khác. Để ghi dữ liệu sử dụng luồng ra, để đọc dữ liệu sử dụng luồng vào

43

4.4. THAO TÁC VỚI FILE VĂN BẢN

Ghi dữ liệu ra file sử dụng StreamWriter

❶ Khai báo và khởi tạo đối tượng **StreamWriter**

```
StreamWriter tên_đối_tượng_streamwriter
 = new StreamWriter("tên_file"[,BooleanAppend]);
```

BooleanAppend = true : ghi nối; = false: ghi đè

❷ Sử dụng phương thức **WriteLine** của đối tượng StreamWriter để copy dữ liệu vào buffer trong bộ nhớ

```
tên_đối_tượng_streamwriter.WriteLine(
                              dữ_liệu_ghi_ra_file);
```

44

4.4. THAO TÁC VỚI FILE VĂN BẢN

❸ Gọi phương thức **Close** của đối tượng StreamWriter để ghi dữ liệu từ buffer sang file và giải phóng tài nguyên

```
tên_đối_tượng_streamwriter.Close();
```

45

4.4. THAO TÁC VỚI FILE VĂN BẢN

Đọc dữ liệu từ file sử dụng StreamReader

❶ Khai báo và khởi tạo đối tượng **StreamReader**

```
StreamReader tên_đối_tượng_streamreader
 = new StreamReader("tên_file");
```

❷ Sử dụng phương thức **ReadLine** để đọc dữ liệu. Cần sử dụng vòng lặp và kiểm tra đến cuối file để đọc nhiều bản ghi

- **Phương thức ReadLine**

```
tên_đối_tượng_streamreader.ReadLine();
```

46

4.4. THAO TÁC VỚI FILE VĂN BẢN

- **Kiểm tra đọc đến cuối file dùng phương thức Peek()**
  - Peek() kiểm tra phần tử tiếp theo mà không thực sự đọc
  - Trả về giá trị -1 nếu đã kiểm tra qua phần tử cuối

❸ Đóng luồng sử dụng phương thức **Close** của đối tượng StreamReader

```
tên_đối_tượng_streamreader.Close();
```

47