Dictionary trong C#

Lớp Dictionary<Tkey,TValue> khá giống SortedList, Dictionary được thiết kế với mục đích tăng hiệu quả với tập dữ liệu lớn, phức tạp.

Một đối tượng dữ liệu lưu vào Dictionary dưới dạng cặp key/value, truy cập đến phần tử thông qua key hoặc thông qua vị trí (index) của dữ liệu trong danh sách. Chú ý, không cho phép trùng key.

Các thuộc tính, phương thức trong Dictionary

|  |  |
| --- | --- |
| **Thành viên** | **Diễn giải** |
| Count | Thuộc tính cho biết số phần tử |
| [key] | Indexer truy cập đến phần tử có key |
| Keys | Thuộc tính là danh sách các key |
| Values | Thuộc tính lấy danh sách các giá trị |
| Add(key, value) | Thêm một phần tử vào Dictionary |
| Remove(key) | Xóa phần tử bằng key của nó |
| Clear() | Loại bỏ tất cả các phần tử |
| ContainKey(key) | Kiểm tra có phần tử nào có khóa là key |
| ContainValue(value) | Kiểm tra có phần tử nào có giá trị là value |

Ví dụ, khai báo Dictionary với key kiểu string, giá trị kiểu int - khởi tạo ngay 2 phần tử

public static void testDic()

{

// Khởi tạo với 2 phần tử

Dictionary<string, int> sodem = new Dictionary<string, int>()

{

["one"] = 1,

["tow"] = 2,

};

// Thêm hoặc cập nhật

sodem["three"] = 3;

var keys = sodem.Keys;

foreach (var k in keys)

{

var value = sodem[k];

Console.WriteLine($"{k} = {value}");

}

}

// one = 1

// two = 2

// three = 3

SortedDictionary trong C#

Lớp SortedDictionary<Tkey,TValue> sử dụng, khai báo và khởi tạo ... giống như lớp Dictionary<Tkey,TValue>.

Chỉ cần lưu ý, nếu dùng SortedDictionary thì các các phần tử được lưu và sắp xếp theo key, thích hợp khi tăng tốc chèn, xóa, tìm kiếm theo key.

HashSet trong C#

HashSet là tập hợp danh sách không cho phép trùng giá trị. HashSet<T> khác với các collection khác là nó cung cấp cơ chế đơn giản nhất để lưu các giá trị, nó không chỉ mục thứ tự và các phần tử không sắp xếp theo thứ tự nào. HashSet<T> cung cấp hiệu năng cao cho các tác vụ tìm kiếm, thêm vào, xóa bỏ ...

Một số phương thức trong HashSet

|  |  |
| --- | --- |
| **Thành viên** | **Diễn giải** |
| Count | Thuộc tính cho biết số phần tử |
| Add(T) | Thêm phần tử vào HashSet |
| Remove(T) | Xóa phần tử khỏi HashSet |
| Clear() | Xóa tất cả các phần tử |
| Contains(T) | Kiểm tra xem có phần tử trong HashSet |
| IsSubsetOf(c) | Kiểm tra xem có là tập con của HashSet c |
| IsSupersetOf(c) | Kiểm tra xem có chứa tập HashSet c |
| UnionWith(c) | Hợp với tập HashSet c |
| IntersectWith(c) | Giao với tập HashSet c |
| ExceptWith(c) | Loại bỏ hết các phần tử chung với c |

HashSet>int< hashset1 = new HashSet>int<() {

5,2,3,4

};

Console.WriteLine($"Phần tử trong hashset1 {hashset1.Count}");

foreach (var v in hashset1)

{

Console.Write(v + " ");

}

Console.WriteLine();

HashSet>int< hashset2 = new HashSet>int<();

hashset2.Add(3);

hashset2.Add(4);

if (hashset1.IsSupersetOf(hashset2))

Console.WriteLine($"hashset1 là tập chứa hashset2");

/\*

Phần tử trong hashset1 4

5 2 3 4

hashset1 là tập chứa hashset2

\*/