|  |  |
| --- | --- |
|  | **Buổi 7** :  **Ứng dụng LinkedList<T> - LinkedListNode<T>**  **trong C#.net** |

LinkedListNode trong C#

LinkedListNode<T> là lớp biểu diễn NÚT trong LinkedList, các đối tượng của LinkedListNode được tạo ra từ LinkedList. Nó có các thuộc tính sau:

|  |  |
| --- | --- |
| **Thuộc tính** | **Ý nghĩa** |
| List | Thuộc tính - tham chiếu (trỏ) đến LinkedList |
| Value | Thuộc tính - là dữ liệu của Node |
| Next | Thuộc tính - tham chiếu (trỏ) đến NÚT tiếp theo (phía sau) - null thì nó là nút cuối |
| Previous | Thuộc tính - tham chiếu (trỏ) đến NÚT phía trước - null thì nó là nút đầu tiên |

|  |  |  |
| --- | --- | --- |
| Previous | Value  Data | Next |

Truy xuất dữ liệu : value.Data

LinkedList trong C#

Trong thư viện .NET cung cấp lớp LinkedList<T> là loại danh sách liên kết kép (từ đây gọi tắt là danh sách liên kết).

Thư viện :

using System.Collections;

using System.Collections.Generic;

**Danh sách liên kết** là một danh sách liên kết kép, chứa các phần tử mà mỗi phần tử trong được gọi là một nút (Node). Mỗi nút ngoài dữ liệu của nút đó, nó sẽ gồm hai biến - một biến tham chiếu đến Node phía trước, một nút tham chiếu đến nút phía sau.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| First | | | Last | |  | |  | |  | |  | |  | | |  | |  | | |  | |
|  | | |  | |  | |  | |  | |  | |  | | |  | |  | | |  | |
| p | v | n | |  | p | v | n |  | p | v | n |  | p | v | n | |  | p | v | n | |  |

Điều này có nghĩa, có có một nút - có thể lấy được nút phía trước nó - cứ thế cho đến nút đầu tiên, nút đầu, tương tự lấy được nút phía sau và dịch chuyển dần được về cuối.

|  |  |
| --- | --- |
| **Member** | **Ý nghĩa** |
| Count | Số nút trong danh sách |
| First | Nút đầu tiên của danh sách |
| Last | Nút đầu tiên của danh sách |
| AddFirst(T) | Chèn một nút có dữ liệu T vào đầu danh sách |
| AddLast(T) | Chèn một nút có dữ liệu T vào cuối danh sách |
| AddAfter(Node, T) | Chèn nút với dữ liệu T, vào sau nút Node (kiểu LinkedListNode) |
| AddBefore(Node, T) | Chèn nút với dữ liệu T, vào trước nút Node (kiểu LinkedListNode) |
| Clear() | Xóa toàn bộ danh sách |
| Contains(T) | Kiểm tra xem có nút với giá trị dữ liệu bằng T |
| Remove(T) | Xóa nút có dữ liệu bằng T |
| RemoveFirst() | Xóa nút đầu tiên |
| RemoveLast() | Xóa nút cuối cùng |
| Find(T) | Tìm một nút, giá trị trả về là một node p (kiểu LinkeListNode) |

**Ví dụ & Bài tập**

Ví dụ 1:

class Program

{

static void XuatDS(LinkedList<int> ds)

{

Console.WriteLine();

// gọi p là một LinkedListNode, cho p -> node đầu tiên

LinkedListNode<int> p = ds.First;

Console.Write(" DSLK : ");

while (p != null)

{

Console.Write(p.Value + " -> ");

p = p.Next;

}

}

static void Main(string[] args)

{

// Khởi tạo dslk kiểu int, rỗng

LinkedList<string> dslk = new LinkedList<string>();

// Khởi tạo dslk kiểu int lấy phần tử array a cùng kiểu

int[] a = { 15, 12, 9, 13, 6, 21, 27, 36 };

LinkedList<int> ds = new LinkedList<int>(a);

// Duyệt ds và in ra

XuatDS(ds);

// thuoc tinh count

Console.WriteLine(" So phan tu : " + ds.Count);

// Thêm vào đầu ds phần tử 39

ds.**AddFirst**(39);

// Thêm vào sau ds phan tu 79

ds.**AddLast**(79)

// Tìm phần tử 36

if (ds.**Find**(36) != null) Console.WriteLine(" 36 co trong ds");

// thêm 63 ngay trước 36

ds.**AddBefore**(ds.**Find**(36), 63);

// lấy "địa chỉ" node p của 36, thêm 33 ngay sau p(36)

LinkedListNode<int> p = ds.**Find**(36);

ds.**AddAfter**(p, 33);

XuatDS(ds);

Console.WriteLine();

// tìm xem có 13 trong ds không?

if (ds.**Contains**(13)) Console.WriteLine(" co 13 trong ds");

// Xóa phần tử 33

ds.**Remove**(33);

// Xóa phần tử đầu ds

ds.**RemoveFirst**();

// Xóa phầ tử sau cùng ds

ds.**RemoveLast**();

XuatDS(ds);

}

}

Ví dụ 2: Định nghĩa DSLK kép DS\_Phanso với mỗi phần tử là các đối tượng Phanso ( gồm tử số và mẫu số). Viết các phương thức cần thiết để giải quyết các thao tác :

* Nhập một phân số vào danh sách (các phân số nhập trước thì xuất trước)
* Xuất các phần tử danh sách dưới dạng biểu thức và tính giá trị (tổng của các phần tử) của nó.

class Phanso

{

int tu;

int mau;

public int Tu { get => tu; set => tu = value; }

public int Mau { get => mau; set => mau = value; }

public Phanso(int t, int m)

{

tu = t; mau = m;

|  |  |  |  |
| --- | --- | --- | --- |
| previous | value | | Next |
| tu | mau |

}

}

class DS\_Phanso

{

LinkedList<**Phanso**> dsps;

public LinkedList<**Phanso**> Dsps { get => dsps; set => dsps = value; }

public DS\_Phanso()

{

dsps = new LinkedList<**Phanso**>();

}

public void Addphanso(int t, int m)

{

**Phanso** ps = new **Phanso**(t, m);

dsps.**AddLast**(ps);

}

public void Outbtps()

{

LinkedListNode<Phanso> p = dsps.**First**;

while(p != null)

{

Console.Write("{0}/{1} -> ", p.**Value**.Tu, p.**Value**.Mau);

p = p.**Next**;

}

}

public double Tong()

{

double t = 0;

LinkedListNode<**Phanso**> p = dsps.**First**;

while (p != null)

{

t = t + (double) p.**Value**.Tu / (double) p.**Value**.Mau;

p = p.**Next**;

}

return t;

}

}

*Câu hỏi lý thuyết* :

1. Viết giải thuật xóa bớt các phần tử trùng lặp trong một danh sách liên kết đơn (không dùng thêm một cấu trúc dữ liệu khác)
2. Để đảo ngược một danh sách liên kết đơn phải làm thế nào? (không dùng thêm một CTDL khác)
3. Làm sao để biết phần tử thứ 3 tính từ phía cuối danh sách liên kết đơn

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

p3 p2 p1

*gọi dslk là danh sách liên kết đơn*

*điều kiện để bài toán có kết quả : dslk.Count >= 3*

*gọi : Node p1 = p2 = p3 = dslk.first*

*while (p1.Next != null)*

*{ p3 = p2; p2 = p1; p1 = p1.Next; }*

*p3 là phần tử thứ 3 tính từ cuối danh sách*

Bài tập : *ứng dụng cấu trúc dữ liệu trong c#.Net*

Trong cuốn từ điển Anh – Việt về thú vật (dành cho trẻ em) mỗi một từ gồm các thông tin :

* en : từ tiếng Anh
* ph : phiên âm quốc tế
* vn : nghĩa tiếng Việt
* loai : loài . Có các loài như sau : 1. Vật nuôi, 2. Động vật hoang dã, 3. côn trùng, 4. Lưỡng cư, 5. Động vật dưới nước,6. Loài chim

Những thao tác khi sử dụng từ điển gồm :

* Tìm từ : với từ tiếng Anh cần tra các thông tin : phiên âm, nghĩa tiếng Việt.
* Tìm loài : với loài đã chọn, thông tin cần tìm là tất cả thú vật của loài gồm : từ en, phiên âm, vn
* Tìm kiếm mở rộng : với từ ban đầu, thì cho biết tất cả các từ mà đầu từ có chứa từ ban đầu

1/ Anh (chị) hãy thiết kế một cấu trúc dữ liệu phù hợp để giải quyết bài toán trên

2/ Cài đặt chương trình (với C#)

Mô hình

Animal.txt

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  | |
| A = 65+0 |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| B = 65+1 |  |  |  |  |  |  |  |  |  |
|  | ... |  |  |  |  |  |  |  |  |  |
| . . . |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| Z = 65+26 |  |  |  |  |  |  |  |  |  |  |

Qui định một từ s thì ký tự đầu là in hoa

Sử dụng array a[26] (có 26 phần tử) mà mỗi a[i] là một danh sách liên kết

a[0] lưu trữ các từ vần ‘A’

a[1] lưu trữ các từ vần ‘B’

. . .

a[26] lưu trữ các từ vần ‘Z’

Một từ s sẽ lưu vào dslk a[s[0] - 65]

Và tìm một từ s sẽ tìm trong dslk a[s[0] - 65]

Danh mục loài được đặt trong array dmloai[6] có 6 phần tử

Một số vấn đề cài đặt

File Animal.txt lưu trữ nội dung có nội dung như đã cho.

Vẽ đường ngang : Console.WriteLine(new string('-', 120));

Nội dung array loài :

string[] Loai = { "1. Vật nuôi", "2. Động vật hoang dã", "3. côn trùng", "4. Lưỡng cư",

"5. Động vật dưới nước","6. Loài chim"};

Đặt tiếng Việt trong màn hình console : Console.OutputEncoding = Encoding.UTF8;

Khi nhập một từ cần tìm 🡪 chuyển ký tự đầu thành in hoa.