Міністерство освіти і науки України

Дніпропетровський національний університет

імені Олеся Гончара

ФФЕКС

Кафедра ЕОМ

**Звіз з лабораторної роботи № 4**

Виконав: студент 2 курсу

Шейко Р.О.

Науковий керівник:

Викладач кафедри EOM

Спірінцева Ольга Володимирівна

м. Дніпро

2022

Предумова. Робота виконувалась на мові с#.

За статею <https://metanit.com/sharp/tutorial/2.16.php>, мова с# має

Ссылочные типы:

* Тип object
* Тип string
* Классы (class)
* Интерфейсы (interface)
* Делегаты (delegate)

Тому за роботу з вказівниками можна вважати за роботу з класами.

Завдання

1. Для організації однозв'язкового списку визначити структурний тип, що містить покажчик на свій тип і поля, які використовуються в роботі №2.
2. Визначити функції вставлення нової ланки в лінійний список, видалення ланки зі списку, перегляд вмісту списку.
3. У функції main() створити ланку списку і перевірити роботу функцій вставки, видалення та перегляду. У функції перегляду передбачено виведення адрес кожної ланки списку. Зробити висновки.
4. Перевести однозв'язковий список у двозв'язковий. Внести відповідні зміни до функцій роботи з списком. Перевірить роботу функції.
5. Перетворити лінійний список на кільцевий. Внести необхідні зміни до функцій роботи зі списком. Перевірити роботу функції. Зробити висновки.

Хід роботи

typedef struct {

char Name[N];

char\* SecondName;

float Number;//nomer//Переменная плаваю- щего типа.

unsigned long SizeOfMassive;//Переменная беззнакового длинного целого типа;

long\* Massive;//Указатель на длинный целый тип;

void\* Behind;

void\* Forward;

} List;

1. Функція вставлення: void Add() класу LinkedList

Функція видалення: void Delete() класу LinkedList

Функція перегляду: void WriteList() класу LinkedList

1. Код до завдання знаходиться у лістингу #1

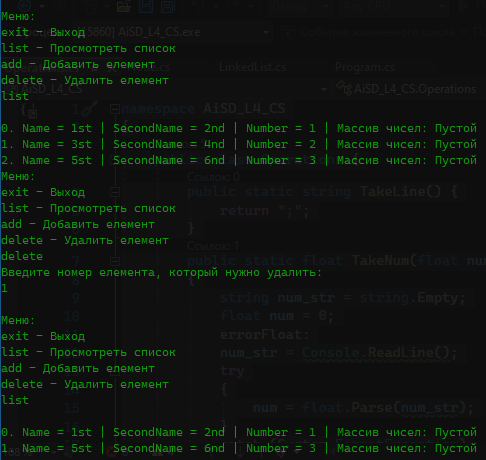


Рис 1. Видалення об’екту

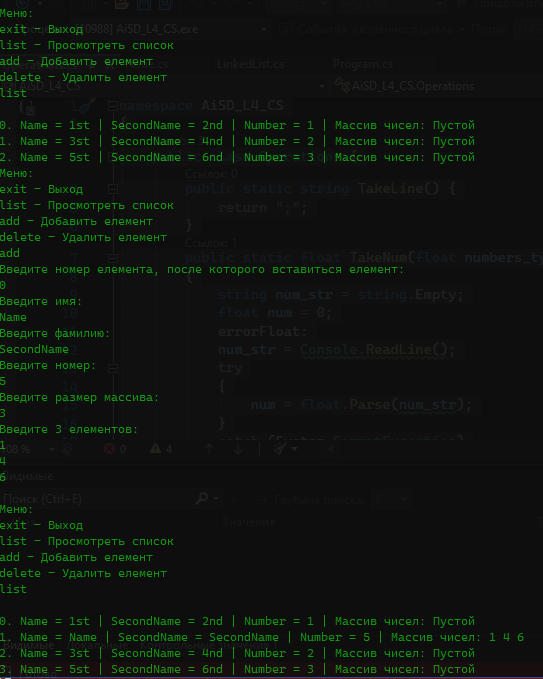


Рис 2. Додавання об’єкту

Висновок: LinkedList дуже класний спосіб для маніпулювання інформацією.

1. Код до завдання знаходиться у лістингу №2

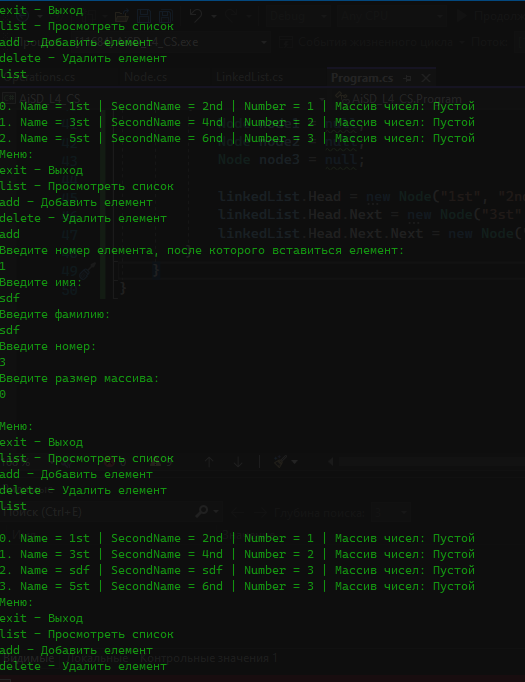


Рис 3. Тестування додавання

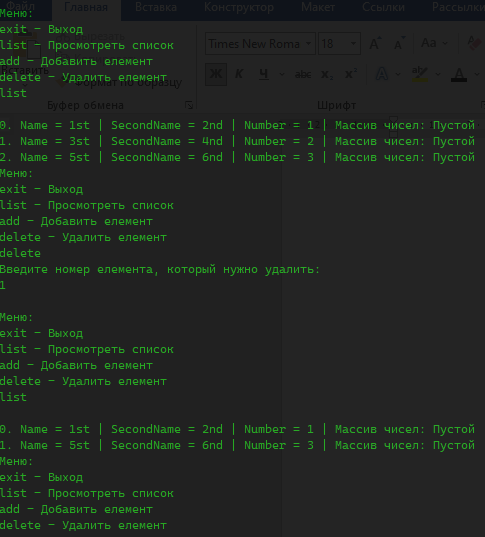


Рис 4. Тестування видалення

1. Код до завдання знаходиться у лістингу №3.

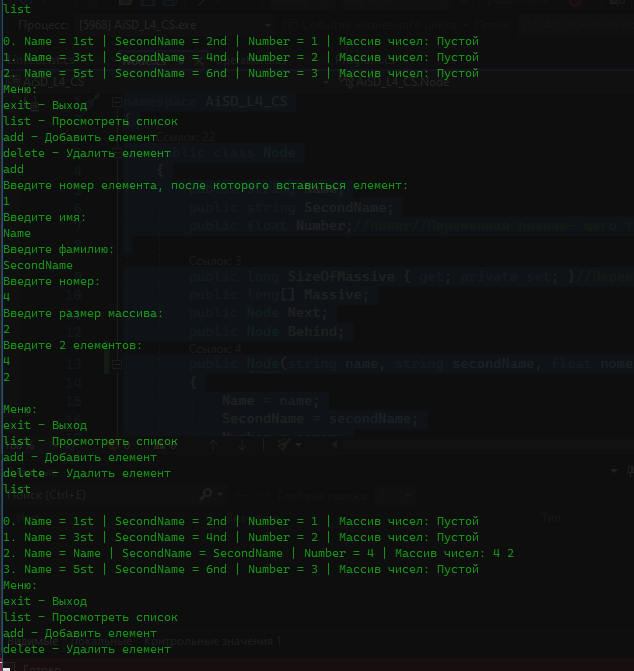


Рис 5. Тестування додавання

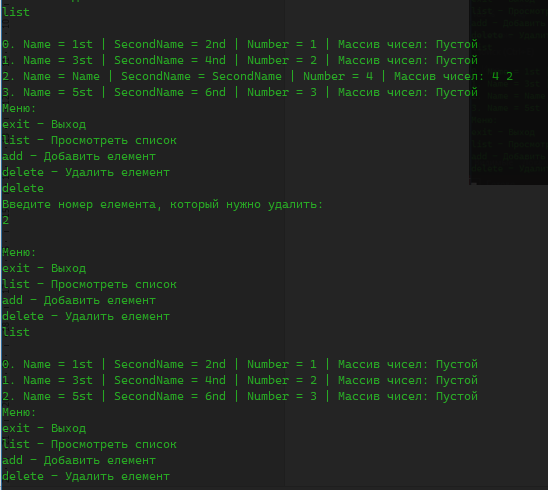


Рис 6. Тестування видалення

Висновок: LinkedList дуже зручний спосіб зберігати інформацію.

**Лістинг №1.**

*Файл “Program.cs”*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace AiSD\_L4\_CS

{

internal class Program

{

static void Main(string[] args)

{

string menu = string.Empty;

string exitTag = "exit";

string listTag = "list";

string addTag = "add";

string deleteTag = "delete";

LinkedList linkedList = new LinkedList();

CreateList(linkedList);

do

{

Console.WriteLine("\nМеню: " +

$"\n{exitTag} - Выход" +

$"\n{listTag} - Просмотреть список" +

$"\n{addTag} - Добавить елемент" +

$"\n{deleteTag} - Удалить елемент");

menu = Console.ReadLine();

if (menu == listTag)

linkedList.WriteList();

else if (menu == addTag)

linkedList.Add();

else if (menu == deleteTag)

linkedList.Delete();

} while (menu != exitTag);

}

private static void CreateList(LinkedList linkedList)

{

linkedList.Head = new Node("1st", "2nd", 1, 0);

linkedList.Head.Next = new Node("3st", "4nd", 2, 0);

linkedList.Head.Next.Next = new Node("5st", "6nd", 3, 0);

}

}

}

*Файл “LinkedList.cs”*

using System.Collections.Generic;

using System.Xml.Linq;

namespace AiSD\_L4\_CS

{

public class LinkedList

{

public Node Head = null;

public void WriteList()

{

Node node = Head;

for (int i = 0; node != null; i++)

{

node.ViewInfo(i);

node = node.Next;

}

}

public void Add() {

string choose = string.Empty;

int chooseInt = 0;

do {

Console.WriteLine("Введите номер елемента, после которого вставиться елемент: ");

chooseInt = Operations.TakeNum(chooseInt);

if (IsHaveElement(chooseInt))

break;

} while (true);

AddElement(TakeNode(chooseInt));

}

private void AddElement(Node nodeBehind)

{

string name = string.Empty;

string secondName = string.Empty;

float number = 0;

long sizeMassive = 0;

Console.WriteLine("Введите имя: ");

name = Console.ReadLine();

Console.WriteLine("Введите фамилию: ");

secondName = Console.ReadLine();

Console.WriteLine("Введите номер: ");

number = Operations.TakeNum(number);

errorMassive: Console.WriteLine("Введите размер массива: ");

sizeMassive = Operations.TakeNum(sizeMassive);

if (sizeMassive < 0) {

Console.WriteLine("Не верный размер!");

goto errorMassive;

}

Node newNode = new Node(name, secondName, number, sizeMassive);

if(sizeMassive > 0)

newNode.Massive = CreateMassive(sizeMassive);

newNode.Next = nodeBehind.Next;

nodeBehind.Next = newNode;

}

private long[] CreateMassive(long sizeMassive) {

long[] massive = new long[sizeMassive];

Console.WriteLine($"Введите {sizeMassive} елементов: ") ;

for(int i = 0; i < sizeMassive; i++)

massive[i] = Operations.TakeNum(massive[i]);

return massive;

}

public void Delete()

{

int chooseInt = 0;

do

{

Console.WriteLine("Введите номер елемента, который нужно удалить: ");

chooseInt = Operations.TakeNum(chooseInt);

if (IsHaveElement(chooseInt))

break;

} while (true);

if(chooseInt > 0)

DeleteElement( TakeNode(chooseInt), TakeNode(chooseInt - 1));

else DeleteElement(TakeNode(chooseInt), this.Head);

}

private void DeleteElement( Node deleteElement, Node behindDeleteElement)

{

behindDeleteElement.Next = deleteElement.Next;

}

private Node TakeNode(int number) {

Node node = Head;

for (int i = 0; i != number; i++ )

node = node.Next;

return node;

}

private bool IsHaveElement(int chooseInt) {

if (chooseInt >= 0 && chooseInt <= CountElements() - 1)

return true;

return false;

}

private int CountElements() {

Node node = Head;

int i = 0;

for (; node != null; i++)

node = node.Next;

return i;

}

}

}

*Файл “Node.cs”*

namespace AiSD\_L4\_CS

{

public class Node

{

public string Name;

public string SecondName;

public float Number;//nomer//Переменная плаваю- щего типа.

public long SizeOfMassive { get; private set; }//Переменная беззнакового длинного целого типа;

public long[] Massive;

public Node Next;

public Node(string name, string secondName, float nomer, long sizeMassive)

{

Name = name;

SecondName = secondName;

Number = nomer;

SizeOfMassive = sizeMassive;

Massive = null;

Next = null;

}

public void ViewInfo(int i) {

Console.Write($"\n{i}. Name = {Name} | SecondName = {SecondName} | Number = {Number} | Массив чисел: ");

for (int j = 0; j < SizeOfMassive; j++)

Console.Write($"{Massive[j]} ");

if (SizeOfMassive == 0)

Console.Write("Пустой");

}

}

}

*Файл “Operations.cs”*

namespace AiSD\_L4\_CS

{

public class Operations {

public static string TakeLine() {

return ";";

}

public static float TakeNum(float numbers\_type)

{

string num\_str = string.Empty;

float num = 0;

errorFloat:

num\_str = Console.ReadLine();

try

{

num = float.Parse(num\_str);

}

catch (System.FormatException)

{

Console.WriteLine("Введен был неверный формат!");

goto errorFloat;

}

return num;

}

public static int TakeNum(int numbers\_type)

{

string num\_str = string.Empty;

int num = 0;

errorInt:

num\_str = Console.ReadLine();

try {

num = Convert.ToInt32(num\_str);

}

catch (System.FormatException) {

Console.WriteLine("Введен был неверный формат!");

goto errorInt;

}

return num;

}

public static long TakeNum(long numbers\_type)

{

string num\_str = string.Empty;

long num = 0;

errorInt:

num\_str = Console.ReadLine();

try

{

num = Convert.ToInt64(num\_str);

}

catch (System.FormatException)

{

Console.WriteLine("Введен был неверный формат!");

goto errorInt;

}

return num;

}

}

}

**Лістинг №2.**

Файл Operations.cs не був змінений, тому його тут немає. Знаходиться у Лістингу №1

*Файл «Program.cs»*

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace AiSD\_L4\_CS

{

internal class Program

{

static void Main(string[] args)

{

string menu = string.Empty;

string exitTag = "exit";

string listTag = "list";

string addTag = "add";

string deleteTag = "delete";

LinkedList linkedList = new LinkedList();

CreateList(linkedList);

do

{

Console.WriteLine("\nМеню: " +

$"\n{exitTag} - Выход" +

$"\n{listTag} - Просмотреть список" +

$"\n{addTag} - Добавить елемент" +

$"\n{deleteTag} - Удалить елемент");

menu = Console.ReadLine();

if (menu == listTag)

linkedList.WriteList();

else if (menu == addTag)

linkedList.Add();

else if (menu == deleteTag)

linkedList.Delete();

} while (menu != exitTag);

}

private static void CreateList(LinkedList linkedList)

{

Node node1 = null;

Node node2 = null;

Node node3 = null;

linkedList.Head = new Node("1st", "2nd", 1, 0, null, node2);

linkedList.Head.Next = new Node("3st", "4nd", 2, 0, node1, node3);

linkedList.Head.Next.Next = new Node("5st", "6nd", 3, 0, null, node2);

}

}

}

*Файл «Node.cs»*

namespace AiSD\_L4\_CS

{

public class Node

{

public string Name;

public string SecondName;

public float Number;//nomer//Переменная плаваю- щего типа.

public long SizeOfMassive { get; private set; }//Переменная беззнакового длинного целого типа;

public long[] Massive;

public Node Next;

public Node Behind;

public Node(string name, string secondName, float nomer, long sizeMassive, Node behind, Node next)

{

Name = name;

SecondName = secondName;

Number = nomer;

SizeOfMassive = sizeMassive;

Massive = null;

Next = next;

Behind = behind;

}

public void ViewInfo(int i) {

Console.Write($"\n{i}. Name = {Name} | SecondName = {SecondName} | Number = {Number} | Массив чисел: ");

for (int j = 0; j < SizeOfMassive; j++)

Console.Write($"{Massive[j]} ");

if (SizeOfMassive == 0)

Console.Write("Пустой");

}

}

}

*Файл «LinkedList.cs»*

using System.Collections.Generic;

using System.Xml.Linq;

namespace AiSD\_L4\_CS

{

public class LinkedList

{

public Node Head = null;

public void WriteList()

{

Node node = Head;

for (int i = 0; node != null; i++)

{

node.ViewInfo(i);

node = node.Next;

}

}

public void Add() {

string choose = string.Empty;

int chooseInt = 0;

do {

Console.WriteLine("Введите номер елемента, после которого вставиться елемент: ");

chooseInt = Operations.TakeNum(chooseInt);

if (IsHaveElement(chooseInt))

break;

} while (true);

AddElement(TakeNode(chooseInt), TakeNode(chooseInt + 1));

}

private void AddElement(Node nodeBehind, Node nodeNext)

{

string name = string.Empty;

string secondName = string.Empty;

float number = 0;

long sizeMassive = 0;

Console.WriteLine("Введите имя: ");

name = Console.ReadLine();

Console.WriteLine("Введите фамилию: ");

secondName = Console.ReadLine();

Console.WriteLine("Введите номер: ");

number = Operations.TakeNum(number);

errorMassive: Console.WriteLine("Введите размер массива: ");

sizeMassive = Operations.TakeNum(sizeMassive);

if (sizeMassive < 0) {

Console.WriteLine("Не верный размер!");

goto errorMassive;

}

Node newNode = new Node(name, secondName, number, sizeMassive, nodeBehind, nodeNext);

if(sizeMassive > 0)

newNode.Massive = CreateMassive(sizeMassive);

newNode.Next = nodeBehind.Next;

nodeBehind.Next = newNode;

}

private long[] CreateMassive(long sizeMassive) {

long[] massive = new long[sizeMassive];

Console.WriteLine($"Введите {sizeMassive} елементов: ") ;

for(int i = 0; i < sizeMassive; i++)

massive[i] = Operations.TakeNum(massive[i]);

return massive;

}

public void Delete()

{

int chooseInt = 0;

do

{

Console.WriteLine("Введите номер елемента, который нужно удалить: ");

chooseInt = Operations.TakeNum(chooseInt);

if (IsHaveElement(chooseInt))

break;

} while (true);

if(chooseInt > 0)

DeleteElement( TakeNode(chooseInt), TakeNode(chooseInt - 1));

else DeleteElement(TakeNode(chooseInt), this.Head);

}

private void DeleteElement( Node deleteElement, Node behindDeleteElement)

{

behindDeleteElement.Next = deleteElement.Next;

}

private Node TakeNode(int number) {

Node node = Head;

for (int i = 0; i != number; i++ )

node = node.Next;

return node;

}

private bool IsHaveElement(int chooseInt) {

if (chooseInt >= 0 && chooseInt <= CountElements() - 1)

return true;

return false;

}

private int CountElements() {

Node node = Head;

int i = 0;

for (; node != null; i++)

node = node.Next;

return i;

}

}

}

**Лістинг №3.**

Файл Operations.cs не був змінений, тому його тут немає. Знаходиться у Лістингу №1.

*Файл “Program.cs”*

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace AiSD\_L4\_CS

{

internal class Program

{

static void Main(string[] args)

{

string menu = string.Empty;

string exitTag = "exit";

string listTag = "list";

string addTag = "add";

string deleteTag = "delete";

LinkedList linkedList = new LinkedList();

CreateList(linkedList);

do

{

Console.WriteLine("\nМеню: " +

$"\n{exitTag} - Выход" +

$"\n{listTag} - Просмотреть список" +

$"\n{addTag} - Добавить елемент" +

$"\n{deleteTag} - Удалить елемент");

menu = Console.ReadLine();

if (menu == listTag)

linkedList.WriteList();

else if (menu == addTag)

linkedList.Add();

else if (menu == deleteTag)

linkedList.Delete();

} while (menu != exitTag);

}

private static void CreateList(LinkedList linkedList)

{

//Add first node.

Node first = new Node("1st", "2nd", 1, 0);

first.Next = null;

first.Behind = null;

//linking with head node

linkedList.Head = first;

//linking next of the node with head

first.Next = linkedList.Head;

//linking prev of the head

linkedList.Head.Behind = first;

//Add second node.

Node second = new Node("3st", "4nd", 2, 0);

second.Next = null;

//linking with first node

second.Behind = first;

first.Next = second;

//linking next of the node with head

second.Next = linkedList.Head;

//linking prev of the head

linkedList.Head.Behind = second;

//Add third node.

Node third = new Node("5st", "6nd", 3, 0);

third.Next = null;

//linking with second node

third.Behind = second;

second.Next = third;

//linking next of the node with head

third.Next = linkedList.Head;

//linking prev of the head

linkedList.Head.Behind = third;

}

}

}

*Файл “LinkedList.cs”*

using System.Collections.Generic;

using System.Xml.Linq;

namespace AiSD\_L4\_CS

{

public class LinkedList

{

public Node Head = null;

public void WriteList()

{

Node node = Head;

Node def = node;

for (int i = 0; i < CountElements(); i++)

{

node.ViewInfo(i);

node = node.Next;

}

}

public void Add() {

string choose = string.Empty;

int chooseInt = 0;

do {

Console.WriteLine("Введите номер елемента, после которого вставиться елемент: ");

chooseInt = Operations.TakeNum(chooseInt);

if (IsHaveElement(chooseInt))

break;

} while (true);

AddElement(TakeNode(chooseInt), TakeNode(chooseInt + 1));

}

private void AddElement(Node nodeBehind, Node nodeNext)

{

string name = string.Empty;

string secondName = string.Empty;

float number = 0;

long sizeMassive = 0;

Console.WriteLine("Введите имя: ");

name = Console.ReadLine();

Console.WriteLine("Введите фамилию: ");

secondName = Console.ReadLine();

Console.WriteLine("Введите номер: ");

number = Operations.TakeNum(number);

errorMassive: Console.WriteLine("Введите размер массива: ");

sizeMassive = Operations.TakeNum(sizeMassive);

if (sizeMassive < 0) {

Console.WriteLine("Не верный размер!");

goto errorMassive;

}

Node newNode = new Node(name, secondName, number, sizeMassive);

if(sizeMassive > 0)

newNode.Massive = CreateMassive(sizeMassive);

nodeBehind.Next.Behind = newNode;

newNode.Next = nodeBehind.Next;

nodeBehind.Next = newNode;

newNode.Behind = nodeBehind;

}

private long[] CreateMassive(long sizeMassive) {

long[] massive = new long[sizeMassive];

Console.WriteLine($"Введите {sizeMassive} елементов: ") ;

for(int i = 0; i < sizeMassive; i++)

massive[i] = Operations.TakeNum(massive[i]);

return massive;

}

public void Delete()

{

int chooseInt = 0;

do

{

Console.WriteLine("Введите номер елемента, который нужно удалить: ");

chooseInt = Operations.TakeNum(chooseInt);

if (IsHaveElement(chooseInt))

break;

} while (true);

if (CountElements() == 1)

{

Console.WriteLine("У нас остался 1 елемент!");

return;

}

if (chooseInt > 0)

DeleteElement(TakeNode(chooseInt), TakeNode(chooseInt - 1));

else {

DeleteElementHead();

DeleteElement(TakeNode(CountElements() - 1), TakeNode(CountElements() - 2));

}

}

private void DeleteElement( Node deleteElement, Node behindDeleteElement)

{

behindDeleteElement.Next = deleteElement.Next;

deleteElement.Next.Behind = behindDeleteElement;

}

private void DeleteElementHead() {

Head.Behind = Head.Next;

Head = Head.Next;

}

private Node TakeNode(int number) {

Node node = Head;

for (int i = 0; i != number; i++ )

node = node.Next;

return node;

}

private bool IsHaveElement(int chooseInt) {

if (chooseInt >= 0 && chooseInt <= CountElements() - 1)

return true;

return false;

}

private int CountElements() {

Node node = Head;

Node def = Head;

int i = 1;

for (; ; i++) {

node = node.Next;

if (node == Head)

break;

}

return i;

}

}

}

*Файл “Node.cs”*

namespace AiSD\_L4\_CS

{

public class Node

{

public string Name;

public string SecondName;

public float Number;//nomer//Переменная плаваю- щего типа.

public long SizeOfMassive { get; private set; }//Переменная беззнакового длинного целого типа;

public long[] Massive;

public Node Next;

public Node Behind;

public Node(string name, string secondName, float nomer, long sizeMassive)

{

Name = name;

SecondName = secondName;

Number = nomer;

SizeOfMassive = sizeMassive;

Massive = null;

Next = null;

Behind = null;

}

public void ViewInfo(int i) {

Console.Write($"\n{i}. Name = {Name} | SecondName = {SecondName} | Number = {Number} | Массив чисел: ");

for (int j = 0; j < SizeOfMassive; j++)

Console.Write($"{Massive[j]} ");

if (SizeOfMassive == 0)

Console.Write("Пустой");

}

}

}