КПІ ім. Ігоря Сікорського
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

Звіт до лабораторної роботи з курсу “Основи програмування. Частина 2. Методології програмування”

Прийняла
доцент кафедри ІПІ
Крамар Ю. М.
“16” квітня 2025 р.

Виконала
Студентка групи ІП-41
Хижняк А. С.

Київ 2025

Лабораторна робота №7

Тема: *Побудова та використання структур даних*

Мета лабораторної роботи *– дослідити типи лінійних та нелінійних структур даних, навчитись користуватись бібліотечними реалізаціями структур даних та будувати власні.*

Завдання:

6	short	Односпрямований	Включення в кінець списку	1.Знайти елемент, кратний заданому значенню. 2.Замінити елементи, що розташовані на парних позиціях, на «0» (нумерація починається з голови списку). 3.Отримати новий список зі значень елементів, значення яких більші за задане значення. 4.Видалити елементи, що розташовані на непарних позиціях (нумерація починається з голови списку).

Текст програми:

```
using System.Collections;

namespace MyList{
    class MyLinkedList : IEnumerable<short>
    {
        private Node head;

        public MyLinkedList()
        {
            head = null;
        }

        public void Add(short data)
        {
            Node newNode = new Node(data);
            if (head == null)
            {
                this.head = newNode;
                return;
            }

            Node current = head;

            while (current.Next != null)
            {
                current = current.Next;
```

```
            }

            current.Next = newNode;
        }

        public void Delete(Node node)
        {
            if (head == null) throw new InvalidOperationException("You cannot delete
element from an empty list");

            if (head == node)
            {
                head = head.Next;
                return;
            }

            Node current = head;
            Node prev = null;

            while (current != node && current != null)
            {
                prev = current;
                current = current.Next;
            }

            if (current == null) return;

            prev.Next = current.Next;
        }

        public MyLinkedList Multiple(short data)
        {
            Node current = head;
            MyLinkedList result = new MyLinkedList();

            while (current != null)
            {
                if((current.Data % data) == 0)
                {
                    result.Add(current.Data);
                }
                current = current.Next;
            }
            return result;
        }

        public void EvenElementsToNull()
        {
            Node current = head;
            int count = 1;
            while (current != null)
```

```
    {
        if((count % 2) == 0)
        {
            current.Data = 0;
        }
        count++;
        current = current.Next;
    }
}

public MyLinkedList GreaterThanGivenNum(short data)
{
    MyLinkedList newList = new MyLinkedList();
    Node current = head;
    while (current != null)
    {
        if(current.Data > data)
        {
            newList.Add(current.Data);
        }
        current = current.Next;
    }
    return newList;
}

public void DeleteOddElements()
{
    Node current = head;
    int count = 1;
    while (current != null)
    {
        if ((count % 2) != 0)
        {
            Delete(current);
        }
        count++;
        current = current.Next;
    }
}

public Node GetHead()
{
    return head;
}

public Node GetByIndex(int index)
{
    if (index < 0) throw new ArgumentOutOfRangeException(nameof(index));

    Node current = head;
    int count = 0;
```

```
            while (current != null)
            {
                if (count == index)
                    return current;

                current = current.Next;
                count++;
            }

            throw new IndexOutOfRangeException("Index out of range!");
        }

        public Node GetByValue(short value)
        {
            Node current = head;

            while (current != null)
            {
                if (current.Data == value)
                    return current;

                current = current.Next;
            }
            throw new InvalidOperationException($"There is no element with data
{value}!");
        }

        public IEnumerator<short> GetEnumerator()
        {
            Node current = head;
            while (current != null)
            {
                yield return current.Data;
                current = current.Next;
            }
        }

        IEnumerator IEnumerable.GetEnumerator()
        {
            return GetEnumerator();
        }
    }
}

class Node
{
    public short Data { get; set; }
    public Node Next { get; set; }

    public Node(short data)
```

```
    {
        this.Data = data;
        this.Next = null;
    }
}

using Microsoft.VisualBasic.FileIO;
using MyList;

class Program
{
    public static void Print(MyLinkedList list)
    {
        Node current = list.GetHead();
        while (current != null)
        {
            Console.Write(current.Data);

            if (current.Next == null) break;

            current = current.Next;
            Console.Write(" -> ");
        }
        Console.WriteLine();
    }


    public static void Main(string[] args)
    {
        MyLinkedList A = new MyLinkedList();
        for (short i = 1; i < 10; i++)
        {
            A.Add(i);
        }

        Console.WriteLine("List A:");
        Print(A);
        Print(A.GreaterThanGivenNum(5));

        foreach (short i in A)
        {
            Console.Write(i + 1 + ", ");
        }
       Console.WriteLine();

        MyLinkedList B = new MyLinkedList();
        Random random = new Random();
        for (int i = 1; i < 20; i++)
        {
            B.Add(Convert.ToInt16(random.Next(1,20)));
        }
```

```
        Console.WriteLine("List B:");
        Print(B);
        B.EvenElementsToNull();
        Print(B);

        MyLinkedList C = new MyLinkedList();
        for (int i = 1; i < 20; i++)
        {
            C.Add(Convert.ToInt16(random.Next(-10, 15)));
        }

        Console.WriteLine("List C:");
        Print(C);
        C.DeleteOddElements();
        Print(C);

        MyLinkedList D = new MyLinkedList();
        for (int i = 1; i < 10; i++)
        {
            D.Add(Convert.ToInt16(random.Next(10, 100)));
        }
        Console.WriteLine("List D:");
        Print(D);
        Print(D.Multiple(1));

        D.Delete(D.GetByIndex(0));
        Print(D);

        Console.WriteLine("List A:");
        A.Delete(A.GetByValue(0));
        Print(A);



    }
}
```

Введені та одержані результати:

```
List A:
1 -> 2 -> 3 -> 4 -> 5 -> 6 -> 7 -> 8 -> 9
6 -> 7 -> 8 -> 9
2, 3, 4, 5, 6, 7, 8, 9, 10,
List B:
3 -> 16 -> 12 -> 16 -> 11 -> 9 -> 5 -> 15 -> 15 -> 5 -> 7 -> 9 -> 5 -> 10 -> 18 -> 7 -> 11 -> 3 -> 6
0 -> 16 -> 0 -> 16 -> 0 -> 9 -> 0 -> 15 -> 0 -> 5 -> 0 -> 9 -> 0 -> 10 -> 0 -> 7 -> 0 -> 3 -> 0
List C:
-2 -> 3 -> 10 -> 5 -> 10 -> 4 -> 6 -> 10 -> 11 -> 2 -> -3 -> -4 -> 10 -> 11 -> 0 -> -4 -> -1 -> 12 -> 7
-2 -> 10 -> 10 -> 6 -> 11 -> -3 -> 10 -> 0 -> -1 -> 7
List D:
75 -> 81 -> 51 -> 56 -> 23 -> 14 -> 10 -> 68 -> 94
75 -> 81 -> 51 -> 56 -> 23 -> 14 -> 10 -> 68 -> 94
```

Теоретичні розрахунки:

В листі А було видалено всі елементи менші 6. В листі В було продемонстровано 2 пункт завдання, а саме парні елементи замінено на «0». В листі С відповіндно реалізовано 3 пункт про видалення непарних елементів. І в останньому листі виведено елементи листа D, що кратні «1».

Висновки: Було досліджено типи структур даних та створено власний односпрямлваний лист. Теоретичні розрахунки відповідають отриманим. Програма працює корректно. Програма вирішує поставлене завдання.