Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки

Кафедра ІПІ(ІСТ)

Звіт

з лабораторної роботи № 7 з дисципліни
«Основи програмування. Частина 2»

„Побудова та використання структур даних”

Виконав(ла) *ІП-43 Дяченко Д.О.*
(шифр, прізвище, ім'я, по батькові)

Перевірив(ла) *Вітковська І.І.*
(прізвище, ім'я, по батькові)

Київ 2025

Мета: дослідити типи лінійних та нелінійних структур даних, навчитись користуватись бібліотечними реалізаціями структур даних та будувати власні.

Варіант 4

№	Тип даних елементів	Тип списку	Спосіб додавання елементу списку	Операції зі списком
4	int	Односпрямований	Включення після першого елементу	1.Знайти перше входження елементу, більшого за задане значення. 2.Знайти суму елементів, значення яких менші за задане значення (нумерація починається з голови списку). 3.Отримати новий список зі значень елементів значення яких більші за задане значення. 4.Видалити елементи, які розташовані після максимального елементу.

Вихідний код програми (мова програмування C#):

```
using System;
using System.Collections;
using System.Collections.Generic;

namespace LinkedListLib
{
   public class Node
   {
      public int Value { get; set; }
      public Node Next { get; set; }
   }

   public class LinkedList : IEnumerable<Node>
   {
      public Node Head;

      public IEnumerator<Node> GetEnumerator()
      {
         Node current = Head;
         while (current != null)
         {
            yield return current;
```

```
            current = current.Next;
        }
    }

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

    public void Print()
    {
        foreach (Node node in this)
        {
            Console.WriteLine(node.Value);
        }
    }

    public void Add(int value)
    {
        Node newNode = new Node() { Value = value };

        if (this.Head == null)
        {
            this.Head = newNode;
        }
        else
        {
            newNode.Next = this.Head.Next;
            this.Head.Next = newNode;
        }
    }

    public Node Find(int value)
    {
        Node currentNode = this.Head;

        while (currentNode != null)
        {
            if (currentNode.Value > value)
            {
                return currentNode;
```

```
            }
            currentNode = currentNode.Next;
        }

        return null;
    }

    public int GetSumm(int value)
    {
        Node currentNode = this.Head;
        int summ = 0;

        while (currentNode != null)
        {
            if (currentNode.Value < value)
            {
                summ += currentNode.Value;
            }
            currentNode = currentNode.Next;
        }

        return summ;
    }

    public LinkedList GetNewList(int value)
    {
        LinkedList newList = new LinkedList();
        Node currentNode = this.Head;

        while (currentNode != null)
        {
            if (currentNode.Value > value)
            {
                newList.Add(currentNode.Value);
            }
            currentNode = currentNode.Next;
        }

        return newList;
    }
```

```
        public void DeleteAfterMax()
        {
            Node currentNode = this.Head;
            Node maxNode = this.Head;

            while (currentNode != null)
            {
                if (currentNode.Value > maxNode.Value)
                {
                    maxNode = currentNode;
                }
                currentNode = currentNode.Next;
            }
            if (maxNode != null)
            {
                maxNode.Next = null;
            }
        }
    }
}
```

```
using System;
using LinkedListLib;

namespace LinkedListApp
{
    class Program
    {
        static void Main(string[] args)
        {
            LinkedList list = new LinkedList();
            list.Add(1);
            list.Add(-2);
            list.Add(8);
            list.Add(-4);
            list.Add(15);
            list.Add(13);

            Console.WriteLine("List:");
            list.Print();
```

```
        Console.WriteLine("\nFirst element bigger than 2:");
        Node found = list.Find(2);
        if (found != null) Console.WriteLine(found.Value);

        Console.WriteLine("\nSum of elements less than 4: " + list.GetSumm(4));

        Console.WriteLine("\nNew list of elements bigger than 3:");
        LinkedList newList = list.GetNewList(3);
        newList.Print();

        Console.WriteLine("\nDeleting elements after the maximum:");
        list.DeleteAfterMax();
        list.Print();
    }
  }
}
```

Висновок: досліджено типи лінійних та нелінійних структур даних. Отрмано теоретичні знання технічної реалізації бібліотечних структур даних. Завдяки отриманим знанням реалізовано односпрямовний цілочисельний список. Усі технічні завдання виконані успішно, програма працює коректно.