using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace BT4

{

internal class Program

{

public class Node

{

public int Data { get; set; }

public Node Next { get; set; }

}

public class MyList

{

private Node head;

private Node tail;

public int Count { get; private set; }

public MyList()

{

head = null;

tail = null;

Count = 0;

}

public void AddFirst(int x)

{

Node newNode = new Node { Data = x };

if (head == null)

{

head = newNode;

tail = newNode;

}

else

{

newNode.Next = head;

head = newNode;

}

Count++;

}

public void AddLast(int x)

{

Node newNode = new Node { Data = x };

if (head == null)

{

head = newNode;

tail = newNode;

}

else

{

tail.Next = newNode;

tail = newNode;

}

Count++;

}

public void ShowList()

{

Node current = head;

while (current != null)

{

Console.Write(current.Data + " ");

current = current.Next;

}

Console.WriteLine();

}

public Node SearchX(int x)

{

Node current = head;

while (current != null)

{

if (current.Data == x)

return current;

current = current.Next;

}

return null;

}

public Node GetMax()

{

Node current = head;

Node maxNode = current;

while (current != null)

{

if (current.Data > maxNode.Data)

maxNode = current;

current = current.Next;

}

return maxNode;

}

public Node GetMin()

{

Node current = head;

Node minNode = current;

while (current != null)

{

if (current.Data < minNode.Data)

minNode = current;

current = current.Next;

}

return minNode;

}

public MyList GetEvenList()

{

MyList evenList = new MyList();

Node current = head;

while (current != null)

{

if (current.Data % 2 == 0)

evenList.AddLast(current.Data);

current = current.Next;

}

return evenList;

}

public MyList GetOddList()

{

MyList oddList = new MyList();

Node current = head;

while (current != null)

{

if (current.Data % 2 != 0)

oddList.AddLast(current.Data);

current = current.Next;

}

return oddList;

}

public void Input()

{

Console.WriteLine("Enter the number of elements:");

int n = Convert.ToInt32(Console.ReadLine());

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

{

Console.WriteLine("Enter element " + (i + 1) + ":");

int x = Convert.ToInt32(Console.ReadLine());

this.AddLast(x);

}

}

}

static void Main(string[] args)

{

MyList myList = new MyList();

int choice, x;

Node node;

do

{

Console.OutputEncoding = Encoding.UTF8;

Console.WriteLine("\nMENU");

Console.WriteLine("1. Thêm một node vào đầu danh sách");

Console.WriteLine("2. Thêm một node vào cuối danh sách");

Console.WriteLine("3. Nhập các giá trị vào danh sách");

Console.WriteLine("4. Hiển thị danh sách");

Console.WriteLine("5. Tìm kiếm một node với giá trị cho trước");

Console.WriteLine("6. Lấy node có giá trị lớn nhất");

Console.WriteLine("7. Lấy node có giá trị nhỏ nhất");

Console.WriteLine("8. Lấy danh sách mới chỉ chứa số chẵn");

Console.WriteLine("9. Lấy danh sách mới chỉ chứa số lẻ");

Console.WriteLine("10. Thoát");

Console.Write("\nNhập lựa chọn của bạn: ");

choice = Convert.ToInt32(Console.ReadLine());

switch (choice)

{

case 1:

Console.Write("\nNhập giá trị để thêm: ");

x = Convert.ToInt32(Console.ReadLine());

myList.AddFirst(x);

break;

case 2:

Console.Write("\nNhập giá trị để thêm: ");

x = Convert.ToInt32(Console.ReadLine());

myList.AddLast(x);

break;

case 3:

myList.Input();

break;

case 4:

myList.ShowList();

break;

case 5:

Console.Write("\nNhập giá trị để tìm kiếm: ");

x = Convert.ToInt32(Console.ReadLine());

node = myList.SearchX(x);

if (node == null)

Console.WriteLine("Không tìm thấy node.");

else

Console.WriteLine("Tìm thấy node.");

break;

case 6:

node = myList.GetMax();

if (node == null)

Console.WriteLine("Danh sách rỗng.");

else

Console.WriteLine("Giá trị lớn nhất là " + node.Data);

break;

case 7:

node = myList.GetMin();

if (node == null)

Console.WriteLine("Danh sách rỗng.");

else

Console.WriteLine("Giá trị nhỏ nhất là " + node.Data);

break;

case 8:

MyList evenList = myList.GetEvenList();

evenList.ShowList();

break;

case 9:

MyList oddList = myList.GetOddList();

oddList.ShowList();

break;

case 10:

Environment.Exit(0);

break;

Default:

Console.WriteLine("\nLựa chọn không hợp lệ. Vui lòng thử lại.");

break;

}

} while (true);

}

}

}