ADI: MERLİN DE L OR

SOYADI: AKENMOE KAMCHE

BÖLÜMÜ: BİLGİSAYAR MÜHENDİSLİĞİ

Soru : bir öğrenci ad, soyad, numara bilgisinin olduğu bir Linked List kodu yazınız.

```
public class StudentNode
{
    public string Name { get; set; }
    public string Surname { get; set; }
    public int Number { get; set; }
    public StudentNode Next { get; set; }

    public StudentNode(string name, string surname, int number)
    {
        Name = name;
        Surname = surname;
        Number = number;
        Next = null;
    }
    public override string ToString()
    {
        return $"Name: {Name}, Surname: {Surname}, Number: {Number}";
    }
}
public class StudentLinkedList
{
    private StudentNode head;

    public StudentLinkedList()
    {
        head = null;
    }
    public bool IsEmpty()
    {
        return head == null;
    }
// Add methods
```

```
public void AddAtBeginning(string name, string surname, int number)
{
    StudentNode newNode = new StudentNode(name, surname, number);
    newNode.Next = head;
    head = newNode;
    Console.WriteLine($"Student {name} {surname} added at beginning.");
}
public void AddAtEnd(string name, string surname, int number)
{
    StudentNode newNode = new StudentNode(name, surname, number);
    if (IsEmpty())
    {
        head = newNode;
    }
    else
    {
        StudentNode current = head;
        while (current.Next != null)
        {
            current = current.Next;
        }
        current.Next = newNode;
    }
    Console.WriteLine($"Student {name} {surname} added at end.");
}
public void AddAfter(int targetNumber, string name, string surname, int number)
{
    StudentNode current = head;
    while (current != null)
    {
        if (current.Number == targetNumber)
        {
            StudentNode newNode = new StudentNode(name, surname, number);
            newNode.Next = current.Next;
            current.Next = newNode;
            Console.WriteLine($"Student added after number {targetNumber}.");
            return;
        }
        current = current.Next;
    }
    Console.WriteLine($"Student with number {targetNumber} not found.");
}
// Delete methods
public void DeleteFromBeginning()
{
    if (IsEmpty())
    {
        Console.WriteLine("List is empty.");
        return;
    }
    Console.WriteLine($"Deleted: {head}");
    head = head.Next;
}
public void DeleteFromEnd()
{
    if (IsEmpty())
    {
        Console.WriteLine("List is empty.");
```

```
            return;
        }
        if (head.Next == null)
        {
            Console.WriteLine($"Deleted: {head}");
            head = null;
            return;
        }
        StudentNode current = head;
        while (current.Next.Next != null)
        {
            current = current.Next;
        }
        Console.WriteLine($"Deleted: {current.Next}");
        current.Next = null;
    }
    public void DeleteByNumber(int number)
    {
        if (IsEmpty())
        {
            Console.WriteLine("List is empty.");
            return;
        }
        if (head.Number == number)
        {
            DeleteFromBeginning();
            return;
        }
        StudentNode current = head;
        while (current.Next != null)
        {
            if (current.Next.Number == number)
            {
                Console.WriteLine($"Deleted: {current.Next}");
                current.Next = current.Next.Next;
                return;
            }
            current = current.Next;
        }
        Console.WriteLine($"Student with number {number} not found.");
    }
    // Display method
    public void Display()
    {
        if (IsEmpty())
        {
            Console.WriteLine("The list is empty.");
            return;
        }
        Console.WriteLine("\n--- Student List ---");
        StudentNode current = head;
        int position = 1;
        while (current != null)
        {
            Console.WriteLine($"{position}. {current}");
            current = current.Next;
            position++;
        }
```

```
        Console.WriteLine("--- End of List ---\n");
    }
    // Search method
    public void SearchByNumber(int number)
    {
        StudentNode current = head;
        while (current != null)
        {
            if (current.Number == number)
            {
                Console.WriteLine($"Student found: {current}");
                return;
            }
            current = current.Next;
        }
        Console.WriteLine($"Student with number {number} not found.");
    }
}
class Program
{
    static void Main(string[] args)
    {
        StudentLinkedList studentList = new StudentLinkedList();

        while (true)
        {
            Console.WriteLine("\n=== Student Linked List Menu ===");
            Console.WriteLine("1. Add student at beginning");
            Console.WriteLine("2. Add student at end");
            Console.WriteLine("3. Add student after a number");
            Console.WriteLine("4. Delete from beginning");
            Console.WriteLine("5. Delete from end");
            Console.WriteLine("6. Delete by number");
            Console.WriteLine("7. Display all students");
            Console.WriteLine("8. Search student by number");
            Console.WriteLine("9. Exit");
            Console.Write("Enter your choice (1-9): ");
            string choice = Console.ReadLine();
            switch (choice)
            {
                case "1":
                    AddStudent(studentList, "beginning");
                    break;
                case "2":
                    AddStudent(studentList, "end");
                    break;
                case "3":
                    if (studentList.IsEmpty())
                    {
                        Console.WriteLine("List is empty. Add students first.");
                        break;
                    }
                    Console.Write("Enter student number to add after: ");
                    if (int.TryParse(Console.ReadLine(), out int targetNumber))
                    {
                        AddStudent(studentList, "after", targetNumber);
                    }
                    else
```

```
                    {
                        Console.WriteLine("Invalid number.");
                    }
                    break;
                case "4":
                    studentList.DeleteFromBeginning();
                    break;
                case "5":
                    studentList.DeleteFromEnd();
                    break;
                case "6":
                    if (studentList.IsEmpty())
                    {
                        Console.WriteLine("List is empty.");
                        break;
                    }
                    Console.Write("Enter student number to delete: ");
                    if (int.TryParse(Console.ReadLine(), out int deleteNumber))
                    {
                        studentList.DeleteByNumber(deleteNumber);
                    }
                    else
                    {
                        Console.WriteLine("Invalid number.");
                    }
                    break;
                case "7":
                    studentList.Display();
                    break;
                case "8":
                    Console.Write("Enter student number to search: ");
                    if (int.TryParse(Console.ReadLine(), out int searchNumber))
                    {
                        studentList.SearchByNumber(searchNumber);
                    }
                    else
                    {
                        Console.WriteLine("Invalid number.");
                    }
                    break;
                case "9":
                    Console.WriteLine("Exiting program. Goodbye!");
                    return;
                default:
                    Console.WriteLine("Invalid choice. Please try again.");
                    break;
            }
        }
    }
    static void AddStudent(StudentLinkedList list, string position, int targetNumber
= 0)
    {
        Console.Write("Enter name: ");
        string name = Console.ReadLine();
        Console.Write("Enter surname: ");
        string surname = Console.ReadLine();
        Console.Write("Enter student number: ");
        if (int.TryParse(Console.ReadLine(), out int number))
```

```
        {
            switch (position)
            {
                case "beginning":
                    list.AddAtBeginning(name, surname, number);
                    break;
                case "end":
                    list.AddAtEnd(name, surname, number);
                    break;
                case "after":
                    list.AddAfter(targetNumber, name, surname, number);
                    break;
            }
        }
        else
        {
            Console.WriteLine("Invalid student number.");
        }
    }
}
```