12214994_SaurabhRana

Title

Student Management System Using LinkedList

Question No : 1 / 1

Console Application: Student Management System Using LinkedList

You are tasked with implementing a **console-based student management system** using a **LinkedList** in C#.
The application should allow users to perform various operations on a collection of **student names**.

The program should be written within the **Program class** in the **Program.cs** file.

Constraints

- All classes must be **public**
- All methods must be **public static**
- Use **LinkedList<string>** to store student names
- Handle user input carefully and display appropriate messages

---

Operations Supported

1. Add Student

- Prompt the user to enter a student's name
- Add the name to the **end of the linked list**
- Display a confirmation message

---

2. Display Students

- Display all student names currently stored in the linked list

---

3. Update Student

- Prompt the user to enter the **current student name**
- If the name exists:
    - Prompt for a **new name**
    - Update the linked list
    - Display a confirmation message
- If not found:
    - Display a message indicating the name was not found

---

4. Delete Student by Name

- Prompt the user to enter the student name to delete
- If the name exists:
    - Remove it from the linked list
    - Display a confirmation message
- If not found:
    - Display a message indicating the name was not found

---

5. Clear List

- Clear all entries in the linked list
- Display a confirmation message

---

6. Exit

- Terminate the program

Instructions

- Display the following menu repeatedly until the user chooses Exit:

```
LinkedList Operations:
1. Add Student
2. Display Students
3. Update Student
4. Delete Student by Name
5. Clear List
6. Exit
```

- Prompt the user to enter a choice between **1–6**
- Execute the corresponding operation

Input Format

- Menu choice (integer 1–6)
- Student names (string) as required by operations

Output Format

- Confirmation messages for add, update, delete, and clear operations
- Display student names line by line
- Error messages when student names are not found
- Exit message on termination

Sample Screenshots (Flow Representation)

Add Student

```
Enter your choice (1-6): 1
Enter student name to add: Deva
Deva added to the list.
```

Display Students

```
Enter your choice (1-6): 2
Students in the list:
Deva
Hari
Suman
```

Update Student

```
Enter your choice (1-6): 3
Enter the current student name to update: Suman
Enter the new student name: Sumanth
Suman updated to Sumanth.
```

If name not found

```
Enter the current student name to update: w
w not found in the list.
```

Delete Student

```
Enter your choice (1-6): 4
Enter student name to delete: Sumanth
Sumanth removed from the list.
```

If name not found

```
Enter student name to delete: gopi
gopi not found in the list.
```

Clear List

```
Enter your choice (1-6): 5
The list has been cleared.
```

Exit

```
Enter your choice (1-6): 6
Exiting...
```

C# Solution (Program.cs)

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

public class Program
{
    public static void AddStudent(LinkedList<string> studentList)
    {
        Console.Write("Enter student name to add: ");
        string name = Console.ReadLine();

        if (string.IsNullOrWhiteSpace(name))
```

```
    {
        Console.WriteLine("Invalid student name.");
        return;
    }

    studentList.AddLast(name);
    Console.WriteLine($"{name} added to the list.");
}

public static void DisplayStudents(LinkedList<string> studentList)
{
    if (studentList.Count == 0)
    {
        Console.WriteLine("No students in the list.");
        return;
    }

    Console.WriteLine("Students in the list:");
    foreach (string student in studentList)
    {
        Console.WriteLine(student);
    }
}

public static void UpdateStudent(LinkedList<string> studentList)
{
    Console.Write("Enter the current student name to update: ");
    string oldName = Console.ReadLine();

    LinkedListNode<string> node = studentList.Find(oldName);

    if (node != null)
    {
        Console.Write("Enter the new student name: ");
        string newName = Console.ReadLine();

        if (string.IsNullOrWhiteSpace(newName))
        {
            Console.WriteLine("Invalid new name.");
            return;
        }

        node.Value = newName;
        Console.WriteLine($"{oldName} updated to {newName}.");
    }
    else
    {
        Console.WriteLine($"{oldName} not found in the list.");
    }
}

public static void DeleteStudentByName(LinkedList<string> studentList)
{
    Console.Write("Enter student name to delete: ");
    string name = Console.ReadLine();

    if (studentList.Remove(name))
```

```
    {
        Console.WriteLine($"{name} removed from the list.");
    }
    else
    {
        Console.WriteLine($"{name} not found in the list.");
    }
}

public static void ClearList(LinkedList<string> studentList)
{
    studentList.Clear();
    Console.WriteLine("The list has been cleared.");
}
public static void Main(string[] args)
{
    LinkedList<string> studentList = new LinkedList<string>();

    while (true)
    {
        Console.WriteLine("LinkedList Operations:");
        Console.WriteLine("1. Add Student");
        Console.WriteLine("2. Display Students");
        Console.WriteLine("3. Update Student");
        Console.WriteLine("4. Delete Student by Name");
        Console.WriteLine("5. Clear List");
        Console.WriteLine("6. Exit");

        Console.Write("Enter your choice (1-6): ");
        string input = Console.ReadLine();

        if (!int.TryParse(input, out int choice))
        {
            Console.WriteLine("Invalid choice.");
            continue;
        }

        switch (choice)
        {
            case 1:
                AddStudent(studentList);
                break;

            case 2:
                DisplayStudents(studentList);
                break;

            case 3:
                UpdateStudent(studentList);
                break;

            case 4:
                DeleteStudentByName(studentList);
                break;

            case 5:
                ClearList(studentList);
```

```
                    break;

                case 6:
                    Console.WriteLine("Exiting...");
                    return;

                default:
                    Console.WriteLine("Invalid choice.");
                    break;
            }
        }
    }

}
```