Rajalakshmi Engineering College

Name: krithika narasimhan
Email: 240701277@rajalakshmi.edu.in
Roll no: 240701277
Phone: 9677451731
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_CY

Attempt : 2
Total Mark : 30
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

Vanessa is learning about the doubly linked list data structure and is eager to play around with it. She decides to find out how the elements are inserted at the beginning and end of the list.

Help her implement a program for the same.

Input Format

The first line of input contains an integer N, representing the size of the doubly linked list.

The next line contains N space-separated integers, each representing the values to be inserted into the doubly linked list.

Output Format

The first line of output prints the integers, after inserting them at the beginning, separated by space.

The second line prints the integers, after inserting at the end, separated by space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
1 2 3 4 5
Output: 5 4 3 2 1
1 2 3 4 5

Answer

```
#include <iostream>
using namespace std;

class Node {
public:
    int data;
    Node* next;
    Node* prev;

    Node(int data) {
        this->data = data;
        this->next = nullptr;
        this->prev = nullptr;
    }
};

class LinkedList {
public:
    Node* head;
    Node* tail;
    int size;

    LinkedList() {
        head = nullptr;
        tail = nullptr;
        size = 0;
    }

    void reverse() {
        Node* current = head;
        Node* temp = nullptr;

        while (current != nullptr) {
            // Swap next and prev pointers for the current node
            temp = current->prev;
            current->prev = current->next;
            current->next = temp;

            // Move to the next node
            current = current->prev;
        }

        // Update head and tail after reversing
        temp = head;
        head = tail;
        tail = temp;
    }

    void push(int new_data) {
        Node* new_node = new Node(new_data);
        new_node->prev = nullptr;
        new_node->next = head;

        if (head != nullptr) {
            head->prev = new_node;
        }

        head = new_node;

        if (size == 0) {
            tail = new_node;
        }

        size++;
    }
```

```
    void printList() {
        Node* current = head;
        while (current != nullptr) {
            cout << current->data << " ";
            current = current->next;
        }
        cout << endl;
    }
};

int main() {
    LinkedList myList;

    int maxSize;
    cin >> maxSize;

    int val;
    for (int i = 0; i < maxSize; i++) {
        cin >> val;
        myList.push(val);
    }

    myList.printList();

    myList.reverse();

    myList.printList();

    return 0;
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

Krishna needs to create a doubly linked list to store and display a sequence of integers. Your task is to help write a program to read a list of integers from input, store them in a doubly linked list, and then display the list.

Input Format

The first line of input consists of an integer n, representing the number of integers in the list.

The second line of input consists of n space-separated integers.

Output Format

The output prints a single line displaying the integers in the order they were added to the doubly linked list, separated by spaces.

If nothing is added (i.e., the list is empty), it will display "List is empty".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
1 2 3 4 5

Output: 1 2 3 4 5

Answer

```
import java.util.Scanner;

class Node {
   int data;
   Node previous;
   Node next;
}

class DoublyLinkedList {
   private Node head;
   private Node tail;
   private int size;

   public void addNode(int data) {
      Node newNode = new Node();
      newNode.data = data;
      if (head == null) {
         head = tail = newNode;
         head.previous = null;
         tail.next = null;
      } else {
         tail.next = newNode;
         newNode.previous = tail;
         tail = newNode;
         tail.next = null;
      }
      size++;
   }

   public void display() {
      Node current = head;
      if (head == null) {
         System.out.println("List is empty");
         return;
      }
      while (current != null) {
         System.out.print(current.data + " ");
         current = current.next;
      }
      System.out.println();
   }

   public static void main(String[] args) {
      DoublyLinkedList list = new DoublyLinkedList();
      Scanner scanner = new Scanner(System.in);
      int n = scanner.nextInt();
      for (int i = 0; i < n; i++) {
         int data = scanner.nextInt();
         list.addNode(data);
      }
      scanner.close();
      list.display();
   }
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Sam is learning about two-way linked lists. He came across a problem where he had to populate a two-way linked list and print the original as well as the reverse order of the list. Assist him with a suitable program.

Input Format

The first line of input consists of an integer n, representing the number of elements in the list.

The second line consists of n space-separated integers, representing the elements.

Output Format

The first line displays the message: "List in original order:"

The second line displays the elements of the doubly linked list in the original order.

The third line displays the message: "List in reverse order:"

The fourth line displays the elements of the doubly linked list in reverse order.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
1 2 3 4 5

Output: List in original order:
1 2 3 4 5
List in reverse order:
5 4 3 2 1

Answer

```
import java.util.Scanner;

class Node {
   int data;
   Node prev;
```

```
    Node next;

    Node(int d) {
        data = d;
        prev = null;
        next = null;
    }
}

class DoublyLinkedList {
    Node head;

    DoublyLinkedList() {
        head = null;
    }

    void insertAtEnd(int data) {
        Node newNode = new Node(data);
        if (head == null) {
            head = newNode;
        } else {
            Node temp = head;
            while (temp.next != null) {
                temp = temp.next;
            }
            temp.next = newNode;
            newNode.prev = temp;
        }
    }

    void displayForward() {
        System.out.println("List in original order:");
        Node temp = head;
        while (temp != null) {
            System.out.print(temp.data + " ");
            temp = temp.next;
        }
        System.out.println();
    }

    void displayReverse() {
        System.out.println("List in reverse order:");
        Node temp = head;
        while (temp.next != null) {
            temp = temp.next;
        }
        while (temp != null) {
            System.out.print(temp.data + " ");
            temp = temp.prev;
        }
        System.out.println();
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        DoublyLinkedList list = new DoublyLinkedList();
        int n = sc.nextInt();
        for (int i = 0; i < n; i++) {
            int data = sc.nextInt();
            list.insertAtEnd(data);
        }
        list.displayForward();
        list.displayReverse();
        sc.close();
    }
}
```

Status : Correct ***Marks : 10/10***