using System;

  public class PrimeNumberExample

   {

     public static void Main(string[] args)

      {

          int n, i, m=0, flag=0;

          Console.Write("Enter the Number to check Prime: ");

          n = int.Parse(Console.ReadLine());

          m=n/2;

          for(i = 2; i <= m; i++)

          {

           if(n % i == 0)

            {

             Console.Write("Number is not Prime.");

             flag=1;

             break;

            }

          }

          if (flag==0)

           Console.Write("Number is Prime.");

     }

   }

**Prime number**

{

    if (isEmpty())

    {

        throw exListEmpty;

    }

    Node target = current;

    Node prevNode = target.prev;

    Node nextNode = target.next;

    // current node is at head

    if (prevNode == null)

    {

        nextNode.prev = null;

        head = nextNode;

    }

    else

    {

        prevNode.next = nextNode;

        current = prevNode;

    }

    // current node is at tail

    if (nextNode == null)

    {

        prevNode.next = null;

        tail = prevNode;

    }

    else

    {

        nextNode.prev = prevNode;

        current = nextNode;

    }

    if (prevNode == null && nextNode == null)

    {

        isEmpty();

    }

    else if (prevNode != null && nextNode != null)

    {

        prevNode.next = nextNode;

        nextNode.prev = prevNode;

        current = nextNode;

    }

    size--;

}

**Remove by using Method**

using System;

public class LinkedList {

    static Node head;

    class Node {

        public int data;

        public Node next, prev;

        public Node(int d)

        {

            data = d;

            next = prev = null;

        }

    }

    /\* Function to reverse a Doubly Linked List \*/

    void reverse()

    {

        Node temp = null;

        Node current = head;

        /\* swap next and prev for all nodes of

        doubly linked list \*/

        while (current != null) {

            temp = current.prev;

            current.prev = current.next;

            current.next = temp;

            current = current.prev;

        }

        /\* Before changing head, check for

          the cases like empty list and

         list with only one node \*/

        if (temp != null) {

            head = temp.prev;

        }

    }

    /\* UTILITY FUNCTIONS \*/

    /\* Function to insert a node at the

    beginning of the Doubly Linked List \*/

    void push(int new\_data)

    {

        /\* allocate node \*/

        Node new\_node = new Node(new\_data);

        /\* since we are adding at the beginning,

        prev is always NULL \*/

        new\_node.prev = null;

        /\* link the old list off the new node \*/

        new\_node.next = head;

        /\* change prev of head node to new node \*/

        if (head != null) {

            head.prev = new\_node;

        }

        /\* move the head to point to the new node \*/

        head = new\_node;

    }

    /\* Function to print nodes in a given

    doubly linked list This function is

    same as printList() of singly linked list \*/

    void printList(Node node)

    {

        while (node != null) {

            Console.Write(node.data + " ");

            node = node.next;

        }

    }

    // Driver code

    public static void Main(String[] args)

    {

        LinkedList list = new LinkedList();

        /\* Let us create a sorted linked list

        to test the functions Created linked

        list will be 10->8->4->2 \*/

        list.push(2);

        list.push(4);

        list.push(8);

        list.push(10);

        Console.WriteLine("Original linked list ");

        list.printList(head);

        list.reverse();

        Console.WriteLine("");

        Console.WriteLine("The reversed Linked List is ");

        list.printList(head);

    }

}

**Reverse a double LL**