Question I: Arrays [30 points]

Create an int array of size 15, and fill in the following 10 integers: [25, 14, 56, 15, 36, 56, 77, 18, 29, 49]

Hints

shifting items: remember to shift the needed items when implementing the below operations.

Tasks

a. insert item 3 at the end of the array. What is its index?
b. insert item 21 at index 0
c. print the item at index 5
d. find the index of item 56
e. delete item 56

Question II: Linked Lists [70 points]

Implement a Linked List satisfying the specifications below

Tasks & Specifications

a. A Linked List consists of a sequence of Node objects
b. A Node object holds a value (assume the value is an Integer type), and a pointer to the next Node: define properties "Value" and "Next" in your Node class accordingly.
c. The LinkedList object will always point to
    a. a Head Node: first Node in the list
    b. a Tail Node: last Node in the list
d. What is the value of Head and Tail when there are zero Nodes in the List? When there is one Node? Are there any problems that could arise under these circumstances?
e. Implement a LinkedList function public **void Append (int Value)** that appends a new Node at the end of the List.
f. Implement a LinkedList function public int **SizeSlow ()** that returns the number of Nodes in a list by traversing the list from its Head till the end.
g. How do you know you have reached the end of the List while traversing it?
h. In a List of size N, SizeSlow needs to traverse N Nodes before being able to answer. Implement a LinkedList function public **int Size ()** that returns the List's size without having to traverse every Node in it
i. Implement a LinkedList function public **void Remove (int Index)** that removes the Node at that index. What if the requested index exceeds the size of the List?
j. Implement a LinkedList function public **int GetValue ()** that returns the Integer value inside the Node at that index.
k. Implement a LinkedList function public **void Insert (int Value, int Index)** that creates a new Node at the given index, filled with the given value.

Question III: BONUS [10 points]

Write a Java method called **areElementsEven** which takes a Node parameter and check if the elements of a linked list are even.

To check if the elements of a linked list are even

For example, if the list was: List1 head→12 →4 →20 →22 →|||

The method returns true

if the list was:

List1 head→12 →13 →21 →20 →|||

The method returns false