Prompt | Scratchpad | Our Solution(s) | Video Explanation | Run Code

Solution 1

```java
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

class Program {
  static class DoublyLinkedList {
    public Node head;
    public Node tail;

    // O(1) time | O(1) space
    public void setHead(Node node) {
      if (head == null) {
        head = node;
        tail = node;
        return;
      }
      insertBefore(head, node);
    }

    // O(1) time | O(1) space
    public void setTail(Node node) {
      if (tail == null) {
        setHead(node);
        return;
      }
      insertAfter(tail, node);
    }

    // O(1) time | O(1) space
    public void insertBefore(Node node, Node nodeToInsert) {
      if (nodeToInsert == head && nodeToInsert == tail) return;
      remove(nodeToInsert);
      nodeToInsert.prev = node.prev;
      nodeToInsert.next = node;
      if (node.prev == null) {
        head = nodeToInsert;
      } else {
        node.prev.next = nodeToInsert;
      }
      node.prev = nodeToInsert;
    }

    // O(1) time | O(1) space
    public void insertAfter(Node node, Node nodeToInsert) {
      if (nodeToInsert == head && nodeToInsert == tail) return;
      remove(nodeToInsert);
      nodeToInsert.prev = node;
      nodeToInsert.next = node.next;
      if (node.next == null) {
        tail = nodeToInsert;
      } else {
        node.next.prev = nodeToInsert;
      }
      node.next = nodeToInsert;
    }

    // O(p) time | O(1) space
    public void insertAtPosition(int position, Node nodeToInsert) {
      if (position == 1) {
        setHead(nodeToInsert);
        return;
      }
      Node node = head;
      int currentPosition = 1;
      while (node != null && currentPosition++ != position) node = nc
      if (node != null) {
        insertBefore(node, nodeToInsert);
      } else {
        setTail(nodeToInsert);
      }
    }

    // O(n) time | O(1) space
    public void removeNodesWithValue(int value) {
      Node node = head;
      while (node != null) {
        Node nodeToRemove = node;
        node = node.next;
        if (nodeToRemove.value == value) remove(nodeToRemove);
```

Your Solutions | Run Code

Solution 1 | Solution 2 | Solution 3

```java
// Feel free to add new properties and methods to the class.
class Program {
  static class DoublyLinkedList {
    public Node head;
    public Node tail;

    public void setHead(Node node) {
      // Write your code here.
    }

    public void setTail(Node node) {
      // Write your code here.
    }

    public void insertBefore(Node node, Node nodeToInsert) {
      // Write your code here.
    }

    public void insertAfter(Node node, Node nodeToInsert) {
      // Write your code here.
    }

    public void insertAtPosition(int position, Node nodeToInsert) {
      // Write your code here.
    }

    public void removeNodesWithValue(int value) {
      // Write your code here.
    }

    public void remove(Node node) {
      // Write your code here.
    }

    public boolean containsNodeWithValue(int value) {
      // Write your code here.
      return false;
    }
  }

  // Do not edit the class below.
  static class Node {
    public int value;
    public Node prev;
    public Node next;

    public Node(int value) {
      this.value = value;
    }
  }
}
```

Custom Output | Raw Output | Submit Code

```
78        }
79      }
80
81      // O(1) time | O(1) space
82      public void remove(Node node) {
83        if (node == head) head = head.next;
84        if (node == tail) tail = tail.prev;
85        removeNodeBindings(node);
86      }
87
88      // O(n) time | O(1) space
89      public boolean containsNodeWithValue(int value) {
90        Node node = head;
91        while (node != null && node.value != value) node = node.next;
92        return node != null;
93      }
94
95      public void removeNodeBindings(Node node) {
96        if (node.prev != null) node.prev.next = node.next;
97        if (node.next != null) node.next.prev = node.prev;
98        node.prev = null;
99        node.next = null;
100     }
101   }
102
103   static class Node {
104     public int value;
```

Run or submit code when you're ready.