```
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None


class LinkedList:

    # initial_elements: allow the collection to start with some elements
    def __init__(self, initial_elements=None):
        self._head = None
        self._count = 0

        if initial_elements:
            for element in initial_elements:
                self.append(element)

    # return a str of the collection
    def __str__(self):
        elements = []
        current = self._head

        while current:
            elements.append(str(current.data))
            current = current.next

        return "[" + ", ".join(elements) + "]"

    # return the length of the elements in the collection
```

```
def __len__(self):
    return self._count

# return the element of the collection in the index position
# Error: the index dont exist
def getitem(self, index):
    if index < 0 or index >= self._count:
        raise IndexError("Index out of range")

    current = self._head
    for _ in range(index):
        current = current.next

    return current.data

# return a boolean that implies if the collection is empty or not
def isEmpty(self):
    return self._count == 0

# allow the collection to be called in a for loop
def __iter__(self):
    current = self._head
    while current:
        yield current.data
        current = current.next

# return a boolean value representing the existence of an element in the collection
def __contains__(self, element):
    current = self._head
```

```
    while current:
        if current.data == element:
            return True
        current = current.next
    return False

# add the element to the end of the collection
def append(self, element):
    new_node = Node(element)

    if self._head is None:
        self._head = new_node
    else:
        current = self._head
        while current.next:
            current = current.next
        current.next = new_node

    self._count += 1

# add the element to the collection at the requested index
# Error: non existing index in the collection
def insert(self, index, element):
    if index < 0 or index > self._count:
        raise IndexError("Index out of range")

    new_node = Node(element)

    if index == 0:
```

```
            new_node.next = self._head
            self._head = new_node
        else:
            current = self._head
            for _ in range(index - 1):
                current = current.next

            new_node.next = current.next
            current.next = new_node

        self._count += 1

    # remove an element in the collection by its value
    # Error: the element dont exist in the collection
    def remove(self, element):
        current = self._head
        previous = None

        while current:
            if current.data == element:
                if previous is None:
                    self._head = current.next
                else:
                    previous.next = current.next

                self._count -= 1
                return

            previous = current
```

```
        current = current.next

    raise ValueError("Element not found")

# remove and return the element in the collection by its index
def pop(self, index):
    if index < 0 or index >= self._count:
        raise IndexError("Index out of range")

    current = self._head
    previous = None

    for _ in range(index):
        previous = current
        current = current.next

    if previous is None:
        self._head = current.next
    else:
        previous.next = current.next

    self._count -= 1
    return current.data

# remove all elements in the collection
def clear(self):
    self._head = None
    self._count = 0
```