Question_3.py U

Lab06_Questions > Question_3.py > DoublyLinkedList > insert

```python
class DLNode:
    def __init__(self, val):
        self.data = val
        self.next = None
        self.prev = None

class DoublyLinkedList:
    def __init__(self):
        self.head = DLNode(None)
        self.count = 1

    def insert(self, val, index):

        newNode = DLNode(val)

        if index > self.count:
            lastNode = self.head
            while lastNode.next is not None:
                lastNode = lastNode.next
            lastNode.next = newNode
            newNode.prev = lastNode
            self.count += 1
            return

        if index == 0:
            newNode.next = self.head
            self.head.prev = newNode
            self.head = newNode
            self.count += 1
            return

        cursor, i = self.head, 0
        while i < index - 1:
            if cursor.next is None:
                break
            cursor = cursor.next
            i += 1
```

Question_3.py U

LabWork > Lab06_Questions > Question_3.py > DLNode

```python
class DoublyLinkedList:
    def insert(self, val, index):

        self.count += 1

    def delete(self, val):
        index, node = self.search(val)
        if node is None:
            return

        p = node.prev
        r = node.next

        if p:
            p.next = r

        if r:
            r.prev = p

        if node == self.head:
            self.head = r

        self.count -= 1

    def traverse(self):
        a = self.head
        while a and a.prev:
            a = a.prev
        while a:
            print(f"{a.data} <--> ", end="")
            a = a.next
        print("None")

    def search(self, val):
        N, i = self.head, 0
        while N:
            if N.data == val:
```

Question_3.py U

LabWork > Lab06_Questions > Question_3.py > DoublyLinkedList > search

```python
class DoublyLinkedList:
    def search(self, val):
            if N.data == val:
                return i, N
            N = N.next
            i += 1
        return None, None

d = DoublyLinkedList()
d.insert('b', 1) # Inserting 'b' at index 1
d.insert('a', 1) #Inserting 'a' at index 1, pushing 'b' to index 2
d.insert('c', 4) # Inserting 'c' at the end (index greater than count,

print("\nAfter Insertion:")
d.traverse()

d.delete('c')
print("\nAfter Deletion of 'c':")
d.traverse()

i, node = d.search('a')
print(f"\nSearch result: 'a' found at index {i}, node = {node.data if
```

OUTPUT

```
PS C:\Coding (VScode)\Codes\University_Related\DSA_NED_LabWork> python

After Insertion:
None <--> a <--> b <--> c <--> None

After Deletion of 'c':
None <--> a <--> b <--> None

Search result: 'a' found at index 1, node = a

PS C:\Coding (VScode)\Codes\University_Related\DSA_NED_LabWork>
```

Question_3.py U

Lab06_Questions > Question_3.py > DoublyLinkedList > insert

```python
class DLNode:
    def __init__(self, val):
        self.data = val
        self.next = None
        self.prev = None

class DoublyLinkedList:
    def __init__(self):
        self.head = DLNode(None)
        self.count = 1

    def insert(self, val, index):

        newNode = DLNode(val)

        if index > self.count:
            lastNode = self.head
            while lastNode.next is not None:
                lastNode = lastNode.next
            lastNode.next = newNode
            newNode.prev = lastNode
            self.count += 1
            return

        if index == 0:
            newNode.next = self.head
            self.head.prev = newNode
            self.head = newNode
            self.count += 1
            return

        cursor, i = self.head, 0
        while i < index - 1:
            if cursor.next is None:
                break
            cursor = cursor.next
            i += 1
```

Question_3.py U

LabWork > Lab06_Questions > Question_3.py > DLNode

```python
class DoublyLinkedList:
    def insert(self, val, index):

        self.count += 1

    def delete(self, val):
        index, node = self.search(val)
        if node is None:
            return

        p = node.prev
        r = node.next

        if p:
            p.next = r

        if r:
            r.prev = p

        if node == self.head:
            self.head = r

        self.count -= 1

    def traverse(self):
        a = self.head
        while a and a.prev:
            a = a.prev
        while a:
            print(f"{a.data} <--> ", end="")
            a = a.next
        print("None")

    def search(self, val):
        N, i = self.head, 0
        while N:
            if N.data == val:
```

Question_3.py U

LabWork > Lab06_Questions > Question_3.py > DoublyLinkedList > search

```python
class DoublyLinkedList:
    def search(self, val):
            if N.data == val:
                return i, N
            N = N.next
            i += 1
        return None, None

d = DoublyLinkedList()
d.insert('b', 1) # Inserting 'b' at index 1
d.insert('a', 1) #Inserting 'a' at index 1, pushing 'b' to index 2
d.insert('c', 4) # Inserting 'c' at the end (index greater than count

print("\nAfter Insertion:")
d.traverse()

d.delete('c')
print("\nAfter Deletion of 'c':")
d.traverse()

i, node = d.search('a')
print(f"\nSearch result: 'a' found at index {i}, node = {node.data if
```

OUTPUT

```
PS C:\Coding (VScode)\Codes\University_Related\DSA_NED_LabWork> python

After Insertion:
None <--> a <--> b <--> c <--> None

After Deletion of 'c':
None <--> a <--> b <--> None

Search result: 'a' found at index 1, node = a

PS C:\Coding (VScode)\Codes\University_Related\DSA_NED_LabWork>
```