Started on	Wednesday, 30 April 2025, 3:25 PM
State	Finished
Completed on	Wednesday, 30 April 2025, 4:01 PM
Time taken	35 mins 49 secs
Grade	**80.00** out of 100.00

Question **1**

Correct

Mark 20.00 out of 20.00

Define a function to delete an element from a specific location in the given linked list.

Answer: (penalty regime: 0 %)

Reset answer

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

class delete_front:
    def __init__(self):
        self.head = None



    def push(self, data):
        if self.head is None:
            self.head = Node(data)
            return
        temp = Node(data)
        temp.next = self.head
        self.head = temp

    def display(self):
        temp1 = self.head
        while temp1 is not None:
```

	Input	Expected	Got	
✔	5 10 20 30 40 50	Enter the number of elements to push: 50 40 30 10	Enter the number of elements to push: 50 40 30 10	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **2**

Correct

Mark 20.00 out of 20.00

Write a python program to print the elements in forward and reverse direction in doubly linked list.

Answer: (penalty regime: 0 %)

Reset answer

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

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

    def push(self, new_data):
        new_node = Node(new_data)
        new_node.next = self.head
        if self.head is not None:
            self.head.prev = new_node
        self.head = new_node

    def append(self, new_data):
        new_node = Node(new_data)
        if self.head is None:
           self.head = new_node
           return
```

	Expected	Got	
✔	Created DLL is: Traversal in forward direction 1 7 6 4 Traversal in reverse direction 4 6 7 1	Created DLL is: Traversal in forward direction 1 7 6 4 Traversal in reverse direction 4 6 7 1	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **3**

Correct

Mark 20.00 out of 20.00

Write a python program to insert an element in the specified position in singly linked list.

Answer: (penalty regime: 0 %)

Reset answer

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

class LinkedList:
    def __init__(self):
        self.head = None

    def traverse_list(self):
        if self.head is None:
            print("List has no element")
            return
        else:
            n = self.head
            while n is not None:
                print(n.data , " ")
                n = n.next

    def insert_at_start(self, data):
        new_node = Node(data)
        new_node.next = self.head
```

	Expected	Got	
✔	After inserting elements at the end 25 35 45 After inserting elements at the beginning 15 25 35 45 Inserting elements at the specific position 15 40 25 35 45	After inserting elements at the end 25 35 45 After inserting elements at the beginning 15 25 35 45 Inserting elements at the specific position 15 40 25 35 45	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.

Question **4**

Not answered

Mark 0.00 out of 20.00

Write a python program to represent strings

For example:

Result
SaveethaEng SaveethaEng Saveetha Eng College

Answer: (penalty regime: 0 %)

```
1
```

Syntax Error(s)

```
Sorry: IndentationError: unexpected indent (__tester__.python3, line 2)
```

Incorrect

Marks for this submission: 0.00/20.00.

Question **5**

Correct

Mark 20.00 out of 20.00

Type a python function to insert element in the doubly linked list in forward and reverse direction.

Answer: (penalty regime: 0 %)

Reset answer

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

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

    def push(self, new_data):
        new_node = Node(new_data)
        new_node.next = self.head
        if self.head is not None:
            self.head.prev = new_node
        self.head = new_node

    def printList(self, node):
        print("\nTraversal in forward direction")
        while node:

            print(node.data)
```

	Expected	Got	
✔	Traversal in forward direction 5 3 1 7 Traversal in reverse direction 7 1 3 5	Traversal in forward direction 5 3 1 7 Traversal in reverse direction 7 1 3 5	✔

Passed all tests! ✔

Correct

Marks for this submission: 20.00/20.00.