BLOCKCHAIN TECHNOLOGY LAB

(20CP406P)

LAB ASSIGNMENT - 3

B.Tech in Computer Science and Engineering Dept.,

Pandit Deendayal Energy University,

Gandhinagar

Name: Mire Kishorkumar Patel

Roll No.: 19BCP080

Branch: Computer Engineering

Lab Assignment 3

Aim: Understanding and Exploring Doubly Linked List

Introduction:

A Doubly Linked List (DLL) contains an extra pointer, typically called the previous pointer, together with the next pointer and data which are there in the singly linked list.

Advantages:

- Deletion of a node is easier than Singly Linked List
- Traversed in both directions- forward and backward

Disadvantages:

- All operations required extra pointer to be maintained
- Requires extra space for previous pointer

Code with the result:

Pandit Deendayal Energy University
School of Technology
Blockchain Technology Lab (20CP406P)
B.Tech-Computer Science & Engineering (Sem-VII)

Name: Mire Kishorkumar Patel
Roll No.: 19BCP080
Branch: Computer Engineering

Lab 3 Assignment
Doubly Linked List

```
# 19BCP080_Mire
# Program: Doubly Linked List

from ctypes import sizeof
from hashlib import sha256
```

```
class node:
    def __init__(self, prev=None, info=None, next=None):
        self.prev = prev
        self.info = info
        self.next = next
```

```
class Doubly_Linked_List:
    def __init__(self):
        self.head = None

    def add_end(self, info):
        new_node = node(info=info)
        if self.head is None:
            self.head = new_node
            return

        last = self.head
        while last.next:
            last = last.next

        last.next = new_node
        new_node.prev = last
        return

    def display_all(self):
        current = self.head
        while current:
            print(current.info)
```

```
            current = current.next
        return

    def add_start(self, info):
        new_node = node(info=info)
        if self.head is None:
            self.head = new_node
            return

        new_node.next = self.head
        self.head.prev = new_node
        self.head = new_node
        return
```

```
x = Doubly_Linked_List()
x.add_end(2)
x.add_end(1)
x.add_end(4)
x.add_start(9)
x.add_end(3)

x.display_all()
```

```
9
2
1
4
3
```