```
#Program to demonstrate singlylinkedlist
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

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

    def append(self, data):  # Insert at the end
        if not self.head:
            self.head = Node(data)
            return
        current = self.head
        while current.next:
            current = current.next
        current.next = Node(data)

    def prepend(self, data):  # Insert at the beginning
        new_node = Node(data)
        new_node.next = self.head
        self.head = new_node

    def insert_after_node(self, prev_node, data):  # Insert after a given
node
        if not prev_node:
            print("Previous node is not in the list.")
            return
        new_node = Node(data)
        new_node.next = prev_node.next
        prev_node.next = new_node

    def delete_node(self, key):  # Delete by value
        current = self.head
        if current and current.data == key:
            self.head = current.next
            current = None
            return
        prev = None
        while current and current.data != key:
            prev = current
            current = current.next
        if current is None:
            return
        prev.next = current.next
        current = None

    def display(self):  # Display the linked list
        current = self.head
        while current:
            print(current.data, end=" ")
            current = current.next
        print()


# Example Usage:
if __name__ == "__main__":
    # Creating a linked list
    linked_list = SinglyLinkedList()

    # Appending elements
    linked_list.append(1)
    linked_list.append(2)
    linked_list.append(3)
    linked_list.append(4)

    # Displaying the linked list
    linked_list.display()

    # Deleting a node
    linked_list.delete_node(3)

    # Displaying the linked list after deletion
    linked_list.display()

    # Prepending an element
    linked_list.prepend(0)

    # Displaying the linked list after prepending
    linked_list.display()

    # Inserting after a node
    node = linked_list.head.next
    linked_list.insert_after_node(node, 2.5)

    # Displaying the linked list after insertion
    linked_list.display()
```

```
****************************************************OUTPUT***************
**************************************************
1 2 3 4
1 2 4
0 1 2 4
0 1 2.5 2 4
```