Q.1. Python code to insert element in array.

Code:

```
print ("Name = Ritesh Dhekane")

arr = [10, 20, 30, 40, 50]

print ("Original Array :", arr)

element = 25
position = 2

arr.append (0)
for i in range (len(arr)-1, position, -1):
        arr[i] = arr[i-1]

array[position = element
print ("Array after insertion :", arr)
```

Output:

```
Name: Ritesh Dhekane
Original Array : [10, 20, 30, 40, 50]
Array after insertion : [10, 20, 25, 30, 40, 50]
```

Q.2. Program to update/modify an element in array at a given index

Code:

```
print ("Name = Ritesh Dhekane")

arr = [10,20,30,40,50]
print("Original Array :", arr)

index = 2
new_value = 99

arr[index] = new_value

print("Array after modification :", arr)
```

Q.3. Search element in array (Linear search)

Code:

```
print("Name = Ritesh Dhekane")

arr = [10, 20, 30, 40, 50]
element = 30
found = -1

for i in range(len(arr)):
    if arr[i] == element:
        found = i
        break

if found != -1:
    print("Element found at index:", found)
else:
    print("Element not found")
```

Q.4. Count elements in array

Code:

```
print("Name = Ritesh Dhekane")

arr = [10,20,30,40,50]

count = 0
for _ in arr:
    count += 1

print("Numbers of elements in array :", count)
```

Q.5. Delete element from array.

Code:

```
print("Name = Ritesh Dhekane")

arr = [10,20,30,40,50]
print("Original Array :", arr)

element = 30
pos = -1

for i in range(len(arr)):
    if arr[i] == element:
        pos = i
        break

if pos != -1
    for i in range(pos, len(arr)-1):
        arr[i] = arr[i+1]
    arr = arr[:-1]
    print("Array after deletion :", arr)
else:
    print("Element not found")
```

Q.6. Sum of all elements

Code:

```
print ("Name: Ritesh Dhekane")

arr = [10,20,30,40,50]
total = 0
for x in arr :
    total += x
print ("Sum of elements :", total)
```

Q.7. Count frequency of each element

Code:

```
print("Name = Ritesh Dhekane")

arr = [10,20,30,10,30,20,10]
freq = {}
for x in arr:
    if x in freq:
        freq[x] += 1
    else:
        freq[x] = 1

print("Frequencies:", freq)
```

Q.8. Average / Mean of array elements

Code:

```
print("Name = Ritesh Dhekane")

arr = [10, 20, 30, 40, 50]
count = 0
total = 0
for x in arr:
    total += x
    count += 1
mean = total / count
print("Average:", mean)
```

Q.9. Merge 2 array

Code:

```
print("Name = Ritesh Dhekane")

arr1 = [1,2,3]
arr2 = [4,5,6]
merged = []

for x in arr1:
    merged.append(x)
for x in arr2:
    merged.append(x)

print("Merged Array:", merged)
```

Q.10. Common element from 2 array

Code

```
print("Name = Ritesh Dhetane")

arr1 = [1,2,3,4,5]
arr2 = [4,5,6,7,8]
common = []

for x in arr1:
    for y in arr2:
        if x==y and x not in common:
            common.append(x)

print("Common elements:", common)
```

Q.11 Find duplicate elements in an array

Code:

```
print("Name = Ritesh Dhekane")

arr = [1,2,3,2,4,5,1]
duplicates = []
for i in range (len(arr)):
    for j in range (i+1, len(arr)):
        if arr[i] == arr[j] and arr[i] not in duplicates:
            duplicates.append(arr[i])
print("Duplicate elements:", duplicates)
```

Q.12. Find maximum and minimum element

Code:

```
print("Name = Ritesh Dhetrane)

arr = [10, 20, 5, 30, 15]
max_val = arr[0]
min_val = arr[0]

for x in arr:
    if x > max_val:
        max_val = x
    if x < min_val:
        min_val = x

print("Maximum element : ", max_val)
print("Minimum element : ", min_val)
```

Q.13. Fibonacci series upto 5 numbers using array

Code:

```
print ("Name = Ritesh Dhekane")

n = 5
fib = [0, 1]

for i in range (2, n):
    fib.append (fib[i-1] + fib[i-2])
print ("Fibonacci Series : ", fib)
```

Page No.
Date

Q.14. Reverse the array.

Code:

```
print("Name = Ritesh Dhekane")

arr = [10, 20, 30, 40, 50]
reversed_arr = []
for i in range(len(arr)-1, -1, -1):
    reversed_arr.append(arr[i])

print("Reversed Array :", reversed_arr)
```

Q.15. Create and display a 2D Array (Matrix)

Code:

```
print(" Name = Ritesh Dhekane ")

rows = 2
cols = 3
matrix = [[0] * cols for _ in range (rows)]

count = 1
for i in range (rows):
    for j in range (cols) :
        matrix [i][j] = count
        count += 1

for row in matrix:
    print (row)
```

Q.16. Matrix Addition

Code:

```
print("Name = Ritesh Dhetrone")

A = [[1,2],[3,4]]
B = [[5,6],[7,8]]

rows = len(A)
cols = len(A[0])
C = [[0]* cols for _ in range(rows)]

for i in range(rows):
    for j in range(cols):
        C[i][j] = A[i][i] + B[i][j]

for row in C:
    print(row)
```

Q.17. Matrix Subtraction

Code:

```
print("Name = Ritesh Dhekane")

A = [[5,6], [7,8]]
B = [[1,2], [3,4]]
rows = len(A)
cols = len(A[0])
C = [[0]*cols for _ in range(rows)]

for i in range(rows):
    for j in range(cols):
        C[i][j] = A[i][j] - B[i][j]

for row in C:
    print(row)
```

Q.18. Matrix Multiplication

Code:

```
print("Name = Ritesh Dhekane")

A = [[1,2],[3,4]]
B = [[5,6],[7,8]]
rows = len(A)
cols = len(B[0])
C = [[0]*cols for _ in range(rows)]

for i in range(rows):
    for j in range(cols):
        for k in range(len(B)):
            C[i][j] += A[i][k] * B[k][j]

for row in C:
    print(row)
```

Q.19. Transpose of Matrix

Code:

```
print ("Name = Ritesh Dhetrane")

A = [[1,2,3],[4,5,6]]
rows = len(A)
cols = len(A[0])
T = [[0]*rows for _ in range(cols)]

for i in range(rows):
    for j in range(cols):
        T[j][i] = A[i][j]

for row in T:
    print(row)
```

Q.20. Sum of Diagonal Elements

Code:

```
print("Name : Ritesh Dhekane")

A = [[1,2,3], [4,5,6], [7,8,9]]
diagno diagonal_sum = 0

for i in range(len(A)):
    diagonal_sum += A[i][i]

print("Sum of diagonal elements : ", diagonal_sum)
```

Q.21. Insert elements at last position of linked list

Code:

```
print("Name = Ritesh Dhekane")

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

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

    def insert_end(self,data):
        new_node = Node(data)
        if not sdf.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next:
            temp = temp.next
        temp.next = new_node

    def display(self):
        temp = self.head
        while temp:
            print(temp.data,end = " ")
            temp = temp.next
        print()

ll = LinkedList()
ll.insert_end(10)
ll.insert_end(20)
ll.insert_end(30)
ll.display()
```

Q.22. Sum of alternate nodes in doubly linked list

Code:

```
print("Name = Ritesh Dhekane")

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

class DoublyLinkedList:
    def __init__(self):
        self.head = None
    def append(self,data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next = new_node
        new_node.prev = temp
    def sum_alternate(self):
        temp = self.head
        sum_alt = 0
        toggle = True
        while temp:
            if toggle:
                sum_alt + temp.data
            toggle = not toggle
            temp = temp.next
        return sum_alt

dll = DoublyLinkedList()
dll.append(10)
dll.append(20)
dll.append(30)
dll.append(40)
print("Sum of alternate nodes:",
        dll.sum_alternate())
```

Q.23. Reverse singly linked list

Code:

```
print("Name = Ritesh Dhekane ")

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

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

    def append(self,data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next:
            temp = temp.next
        temp.next = new_node

    def reverse(self):
        prev = None
        current = self.head
        while current:
            nxt = current.next
            current.next = prev
            prev = current
            current = nxt
        self.head = prev
    def display(self):
        temp = self.head
        while temp:
            print(temp.data, end = " ")
            temp = temp.next
        print()
ll = LinkedList()
ll.append(10)
ll.append(20)
ll.append(30)
ll.reverse()
ll.display()
```

Q. 24. Delete node from beginning in singly linked list

Code:

```
print("Name = Ritesh Dhetkane")

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

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

    def append(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next:
            temp = temp.next
        temp.next = new_node

    def delete_beginning(self):
        if self.head:
            self.head = self.head.next

    def display(self):
        temp = self.head
        while temp:
            print(temp.data, end=" ")
            temp = temp.next
        print()

ll = LinkedList()
ll.append(10)
ll.append(20)
ll.append(30)
ll.delete_beginning()
ll.display()
```

Q.25 Reverse nodes of linked list

Code:

```
print("Ritesh Dhekane")

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

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

    def append(self,data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next:
            temp = temp.next
        temp.next = new_node

    def reverse_nodes(self):
        prev = None
        current = self.head
        while current:
            nxt = current.next
            current.next = prev
            prev = current
            current = nxt
        self.head = prev
    def display(self):
        temp = self.head
        while temp:
            print(temp.data, end=" ")
            temp = temp.next
        print()
ll = LinkedList()
ll.append(1)
ll.append(2)
ll.append(3)
ll.append(4)
ll.reverse_nodes()
ll.display()
```

Q 26. Delete elements from linked list whose sum is zero

Code:

```
print("Name = Ritesh Dhetrane")

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

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

    def append(self,data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next:
            temp = temp.next
        temp.next = new_node

    def delete_zero_sum(self):
        prev = None
        current = self.head
        while current:
            if current.data == 0:
                if prev:
                    prev.next = current.next
                else:
                    self.head = current.next
            else:
                prev = current
            current = current.next
    def display(self):
        temp = self.head
        while temp:
            print(temp.data, end=" ")
            temp = temp.next
        print()
ll = LinkedList()
ll.append(10)
ll.append(0)
ll.append(-5)
ll.append(0)
ll.delete_zero_sum()
ll.display()
```

Q.27. Doubly Linked List and return count of elements

Code:

```
print("Name : Ritesh Dhekane")

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

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

    def append(self,data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next
            temp.next = new_node
            new_node.prev = temp

    def count(self):
        temp = self.head
        cnt = 0
        while temp:
            cnt += 1
            temp = temp.next
        return cnt

dll = DoublyLinkedList()
dll.append(10)
dll.append(20)
dll.append(30)
print("Count of elements:", dll.count())
```

Q.28 Doubly linked list search and return position

Code:

```
print("Name : Ritesh Dhekane")

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

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

    def append(sdf,data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next:
            temp = temp.next
        temp.next = new_node
        new_node.prev = temp

    def search(self, value):
        temp = self.head
        pos = 1
        while temp:
            if temp.data == value:
                return pos
            temp = temp.next
            pos += 1
        return -1

dll = DoublyLinkedList()
dll.append(10)
dll.append(20)
dll.append(30)
print("Position of 20 :",
                    dll.search(20))
```

Q.29 Remove duplicated elements from sorted SLL

Code:

```
print("Name = Ritesh Dhelam")
class Node:
    def __init__(self, data):
        self.data = data
        self.next = None

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

    def append(self, data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next
            temp = temp.next
        temp.next = new_node

    def remove_duplicates(self):
        temp = self.head
        while temp and temp.next:
            if temp.data == temp.next.data:
                temp.next = temp.next.next
            else:
                temp = temp.next
    def display(self):
        temp = self.head
        while temp:
            print(temp.data, end = " ")
            temp = temp.next
        print()
ll = LinkedList()
ll.append(3)
ll.append(3)
ll.append(4)
ll.append(5)
ll.remove_duplicates()
ll.display()
```

Q.30. Insertion and deletion in singly Linked list

Code:

```
print("Name = Ritesh Dhekane")

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

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

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

    def delete_end(self):
        if not self.head:
            return
        if not self.head.next:
            self.head = None
            return
        temp = self.head
        while temp.next.next:
            temp = temp.next
        temp.next = None

    def display(self):
        temp = self.head
        while temp:
            print(temp.data, end=" ")
            temp = temp.next
        print()

ll = LinkedList()
ll.insert_beginning(10)
ll.insert_beginning(20)
ll.insert_beginning(30)
ll.delete_end()
ll.display()
```

Q.31 Count number of nodes in singly linked list

Code:

```
print("Name = Ritesh Dhekane")

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

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

    def append(self,data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next:
            temp = temp.next
        temp.next = new_node

    def count(self):
        temp = self.head
        cnt = 0
        while temp:
            cnt += 1
            temp = temp.next
        return cnt

ll = LinkedList()
ll.append(10)
ll.append(20)
ll.append(30)
print("Number of nodes: ",
      ll.count())
```

Q.32 Merge 2 sorted linked lists.

Code:

```
print("Name = Ritesh Dhekane")

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

def merge_sorted(l1, l2):
    dummy = Node(0)
    tail = dummy
    while l1 and l2:
        if l1.data < l2.data:
            tail.next = l1
            l1 = l1.next
        else:
            tail.next = l2
            l2 = l2.next
        tail = tail.next
    if l1:
        tail.next = l1
    if l2:
        tail.next = l2
    return dummy.next

def print_list(head):
    temp = head
    while temp:
        print(temp.data, end=" ")
        temp = temp.next
    print()

a1 = Node(1)
a1.next = Node(3)
a1.next.next = Node(5)

a2 = Node(2)
a2.next = Node(4)
a2.next.next = Node(6)

merge = merge_sorted(a1, a2)
print_list(merged)
```

Q.33 Insert at specific position in SLL

Code:

```
print("Name = Ritesh Dhetrane")

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

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

    def insert_at_position(self, data, pos):
        new_node = Node(data)
        if pos == 1:
            new_node.next = self.head
            self.head = new_node
            return
        temp = self.head
        for i in range(pos - 2):
            if temp is None:
                return
            temp = temp.next
        new_node.next = temp.next
        temp.next = new_node

    def display(self):
        temp = self.head
        while temp:
            print(temp.data, end=" ")
            temp = temp.next
        print()

ll = LinkedList()
ll.insert_at_position(10,1)
ll.insert_at_position(20,2)
ll.insert_at_position(30,2)
ll.display()
```

Q.34. Delete from Beginning and end in DLL

Code:

```
print("Name = Ritesh Dhekane")

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

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

    def append(self,data):
        new_node = Node(data)
        if not self.head:
            self.head = new_node
            return
        temp = self.head
        while temp.next
            temp = temp.next
        temp.next = new_node
        new_node.prev = temp

    def delete_beginning(self):
        if self.head:
            self.head = self.head.next
            if self.head:
                self.head.prev = None

    def delete_end(self):
        if not self.head:
            return
        if not self.head.next:
            self.head = None
            return
        temp = self.head
        while temp.next:
            temp = temp.next
        temp.prev.next = None

    def display(self)
        temp = self.head
        while temp:
            print(temp.data, end=" ")
            temp = temp.next
        print()

dll = DoublyLinkedList()
dll.append(10)
dll.append(20)
dll.append(30)
dll.delete_beginning()
dll.delete_end()
dll.display()
```

Q.35. Merge 2 sorted Doubly Linked Lists.

Code:

```
print("Name = Ritesh Dhetrane ")

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

def merge_sorted_dll(h1,h2):
    dummy = Node(0)
    tail = dummy
    while h1 and h2:
        if h1.data < h2.data:
            tail.next = h1
            h1.prev = tail
            h1 = h1.next
        else:
            tail.next = h2
            h2.prev = tail
            h2 = h2.next
        tail = tail.next
    while h1:
        tail.next = h1
        h1.prev = tail
        tail = tail.next
        h1 = h1.next
    while h2:
        tail.next = h2
        h2.prev = tail
        tail = tail.next
        h2 = h2.next
    if dummy.next
        dummy.next.prev = None
    return dummy.next

def print_dll(head):
    temp = head
    while temp:
        print(temp.data, end=" ")
        temp = temp.next
    print()

a1 = Node(1)
a1.next = Node(3)
a1.next.prev = a1
a1.next.next = Node(5)
a1.next.next.prev = a1.next

a2 = Node(2)
a2.next = Node(4)
a2.next.prev = a2
a2.next.next = Node(6)
a2.next.next.prev = a2.next
merged = merge_sorted_dll(a1,a2)
print_dll(merged)
```