#### Singly Linked List (singly\_linkedlist.py)

class Node:

def \_\_init\_\_(self, data):

self.data = data

self.next = None

class SinglyLinkedList:

def \_\_init\_\_(self):

self.head = None

def insert\_at\_beginning(self, data):

new\_node = Node(data)

new\_node.next = self.head

self.head = new\_node

def display(self):

current = self.head

while current:

print(current.data, end=" -> ")

current = current.next

print("None")

#### Problem 1 for Singly Linked List (singly\_linkedlist\_problem1.py): Insertion at the Beginning

from singly\_linkedlist import SinglyLinkedList

if \_\_name\_\_ == "\_\_main\_\_":

linked\_list = SinglyLinkedList()

linked\_list.insert\_at\_beginning(10)

linked\_list.insert\_at\_beginning(20)

linked\_list.insert\_at\_beginning(30)

print("Singly Linked List after insertions:")

linked\_list.display()

#### Problem 2 for Singly Linked List (singly\_linkedlist\_problem2.py): Deletion of a Node

class SinglyLinkedList:

def delete\_node(self, key):

current = self.head

if current and current.data == key:

self.head = current.next

current = None

return

prev = None

while current:

if current.data == key:

break

prev = current

current = current.next

if current is None:

print(f"Node with value {key} not found.")

return

prev.next = current.next

current = None

if \_\_name\_\_ == "\_\_main\_\_":

linked\_list = SinglyLinkedList()

linked\_list.insert\_at\_beginning(10)

linked\_list.insert\_at\_beginning(20)

linked\_list.insert\_at\_beginning(30)

print("Original list:")

linked\_list.display()

linked\_list.delete\_node(20)

print("List after deletion:")

linked\_list.display()

#### Singly Linked List with Tail (singly\_linkedlist\_with\_tail.py)

class Node:

def \_\_init\_\_(self, data):

self.data = data

self.next = None

class SinglyLinkedListWithTail:

def \_\_init\_\_(self):

self.head = None

self.tail = None

def insert\_at\_end(self, data):

new\_node = Node(data)

if not self.tail:

self.head = self.tail = new\_node

return

self.tail.next = new\_node

self.tail = new\_node

def display(self):

current = self.head

while current:

print(current.data, end=" -> ")

current = current.next

print("None")

#### Problem 3 for Singly Linked List with Tail (singly\_linkedlist\_problem3.py): Insertion at the End

from singly\_linkedlist\_with\_tail import SinglyLinkedListWithTail

if \_\_name\_\_ == "\_\_main\_\_":

linked\_list = SinglyLinkedListWithTail()

linked\_list.insert\_at\_end(10)

linked\_list.insert\_at\_end(20)

linked\_list.insert\_at\_end(30)

print("Singly Linked List with Tail after insertions:")

linked\_list.display()

#### Problem 4 for Singly Linked List with Tail (singly\_linkedlist\_problem4.py): Deletion from the End

class SinglyLinkedListWithTail:

def delete\_from\_end(self):

if self.head is None:

print("List is empty")

return

if self.head == self.tail:

self.head = self.tail = None

return

current = self.head

while current.next != self.tail:

current = current.next

current.next = None

self.tail = current

if \_\_name\_\_ == "\_\_main\_\_":

linked\_list = SinglyLinkedListWithTail()

linked\_list.insert\_at\_end(10)

linked\_list.insert\_at\_end(20)

linked\_list.insert\_at\_end(30)

print("Original list:")

linked\_list.display()

linked\_list.delete\_from\_end()

print("List after deletion from end:")

linked\_list.display()

#### ****Doubly Linked List**** (doubly\_linkedlist.py)

class Node:

def \_\_init\_\_(self, data):

self.data = data

self.next = None

self.prev = None

class DoublyLinkedList:

def \_\_init\_\_(self):

self.head = None

def insert\_at\_beginning(self, data):

new\_node = Node(data)

if not self.head:

self.head = new\_node

else:

new\_node.next = self.head

self.head.prev = new\_node

self.head = new\_node

def display(self):

current = self.head

while current:

print(current.data, end=" <-> ")

current = current.next

print("None")

#### Problem 1 for Doubly Linked List (doubly\_linkedlist\_problem1.py): Insert at the Beginning

from doubly\_linkedlist import DoublyLinkedList

if \_\_name\_\_ == "\_\_main\_\_":

dll = DoublyLinkedList()

dll.insert\_at\_beginning(10)

dll.insert\_at\_beginning(20)

dll.insert\_at\_beginning(30)

print("Doubly Linked List after insertions:")

dll.display()

#### Problem 2 for Doubly Linked List (doubly\_linkedlist\_problem2.py): Deletion from the Middle

class DoublyLinkedList:

def delete\_node(self, key):

current = self.head

if current and current.data == key:

if current.next:

current.next.prev = None

self.head = current.next

current = None

return

while current:

if current.data == key:

break

current = current.next

if current is None:

print(f"Node with value {key} not found.")

return

if current.next:

current.next.prev = current.prev

if current.prev:

current.prev.next = current.next

current = None

if \_\_name\_\_ == "\_\_main\_\_":

dll = DoublyLinkedList()

dll.insert\_at\_beginning(10)

dll.insert\_at\_beginning(20)

dll.insert\_at\_beginning(30)

print("Original list:")

dll.display()

dll.delete\_node(20)

print("List after deletion:")

dll.display()