**NAMA : Ainul Sholati Hifiati**

**NIM : 24241002**

**MATKUL : STRUKTUR DATA**

**Modul 2** Main. Py class Node:

def \_\_init\_\_(self, data): # Perbaikan di sini self.data = data self.prev = None self.next = None

class DoubleLinkedList:

def \_\_init\_\_(self): # Perbaikan di sini self.head = None

# Tambah node di akhir def append(self, data): new\_node = Node(data) if self.head is None: self.head = new\_node

return curr = self.head while curr.next: curr = curr.next curr.next = new\_node new\_node.prev = curr

# Hapus node awal

def delete\_front(self): if self.head is None: print("List kosong.") return

print(f"Menghapus node awal: {self.head.data}") self.head = self.head.next if self.head:

self.head.prev = None

# Hapus node akhir def delete\_end(self): if self.head is None: print("List kosong.") return curr = self.head while curr.next: curr = curr.next print(f"Menghapus node akhir: {curr.data}")

if curr.prev:

curr.prev.next = None else:

self.head = None

# Hapus node berdasarkan nilai def delete\_by\_value(self, value):

curr = self.head while curr: if curr.data == value:

print(f"Menghapus node dengan nilai: {value}") if curr.prev:

curr.prev.next = curr.next else:

self.head = curr.next if curr.next: curr.next.prev = curr.prev return curr = curr.next print(f"Data {value} tidak ditemukan.")

# Cetak semua data def display(self): curr = self.head while curr:

print(curr.data, end=" <-> ") curr = curr.next print("None")

# Contoh penggunaan dll = DoubleLinkedList() dll.append(10) dll.append(20) dll.append(30) dll.append(40)

print("Linked List Awal:") dll.display()

dll.delete\_front()

dll.display()

dll.delete\_end() dll.display()

dll.delete\_by\_value(20) dll.display()

**outputnya** **:**

Linked List Awal:

10 <-> 20 <-> 30 <-> 40 <-> None

Menghapus node awal: 10

20 <-> 30 <-> 40 <-> None

Menghapus node akhir: 40

20 <-> 30 <-> None

Menghapus node dengan nilai: 20

30 <-> None

**Penjelasannya :**

# Class Node

* Baris ini membuat class Node, yaitu struktur dasar dari list.
* Fungsi \_\_init\_\_ dijalankan saat objek Node dibuat.
* Self.data menyimpan nilai yang ingin ditaruh dalam node.
* Self.prev adalah penghubung ke node sebelumnya (default-nya kosong).
* Self.next adalah penghubung ke node berikutnya (default-nya kosong).

**2. Class DoubleLinkedList**

* Ini adalah **class utama** untuk daftar berantai ganda (*double linked list*).
* Fungsi \_\_init\_\_ membuat list kosong, dengan head menunjuk ke None.

1. **Menambahkan Node di Akhir** 
   * + Membuat node baru dengan data yang diberikan.
     + Jika list masih kosong, node baru langsung jadi kepala (head).
     + Jika tidak kosong, kita cari node terakhir dengan perulangan.
     + Setelah ketemu node terakhir, kita sambungkan node baru ke belakangnya, dan sebaliknya.
2. **Menghapus Node Pertama (Depan)** 
   * + Cek apakah list kosong. Jika ya, tampilkan pesan.
     + Tampilkan nilai yang dihapus, lalu pindahkan head ke node berikutnya.
     + Jika masih ada node setelahnya, putuskan hubungan ke node yang dihapus tadi.
3. **Menghapus Node Terakhir (Belakang)** 
   * + Jika list kosong, tampilkan pesan.
     + Temukan node terakhir dengan perulangan.
     + Tampilkan nilai yang dihapus.
     + Jika node punya sebelumnya, putuskan koneksinya.
     + Jika tidak (hanya 1 node), kosongkan list.
4. **Menghapus Node Berdasarkan Nilai** 
   * + Mulai dari head dan cari node yang cocok.
     + Jika ditemukan, tampilkan bahwa node akan dihapus.
     + Jika bukan node pertama, sambungkan node sebelumnya ke sesudahnya.
     + Jika itu adalah node pertama, geser head.
     + Jika ada node setelahnya, sambungkan ke node sebelumnya.• Keluar dari fungsi setelah menghapus.
     + Jika data tidak ditemukan setelah pencarian selesai, tampilkan pesan.
5. **Menampilkan Isi List**

**•** fungsi ini mencetak semua data dari awal sampai akhir.

* + - Menambahkan panah "<->" sebagai penghubung antar node.
    - Di akhir, mencetak "None" untuk menandai akhir list.

1. **Contoh Penggunaan** 
   * Membuat objek list kosong.
   * Menambahkan 4 data ke dalam list.
   * Menampilkan isi list: 10 <-> 20 <-> 30 <-> 40 <-> None
   * Menghapus data paling depan (10), lalu tampilkan hasilnya.
   * Menghapus data paling belakang (40), lalu tampilkan hasilnya. • Menghapus node dengan nilai 20, lalu tampilkan list yang tersisa.