

NAMA : SUPIARTINI

NIM : 24241009

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 :

1. 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.

3. 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.

4. 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.

5. 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.

6. 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.

7. 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.

8. 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 nod