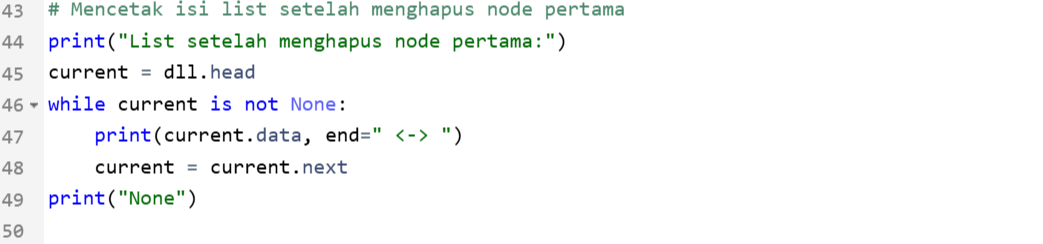
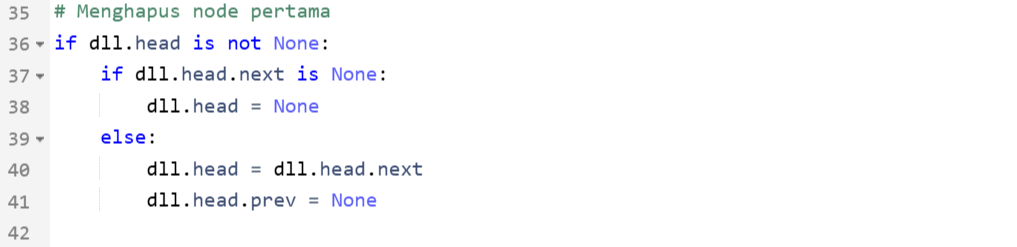
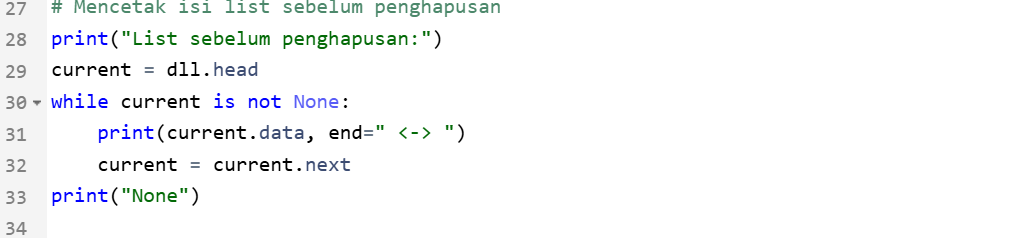
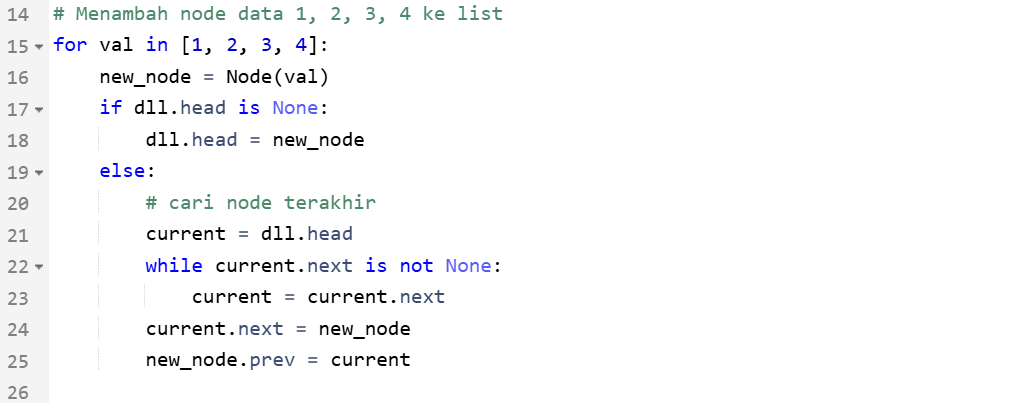
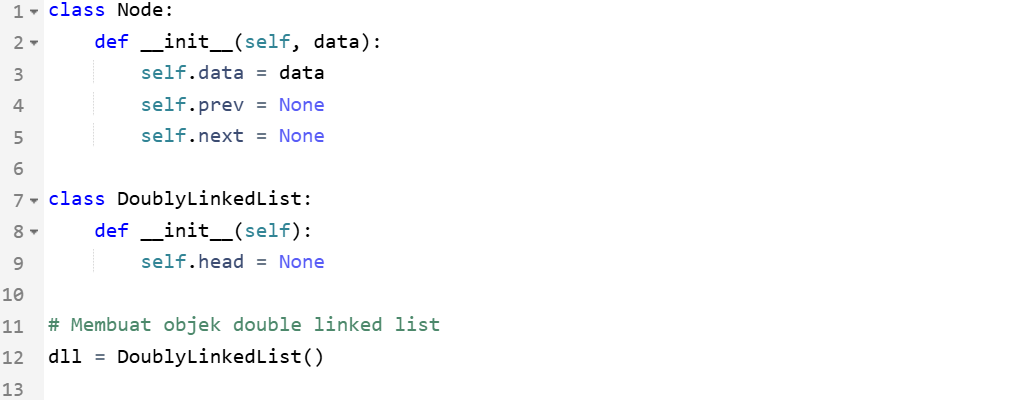
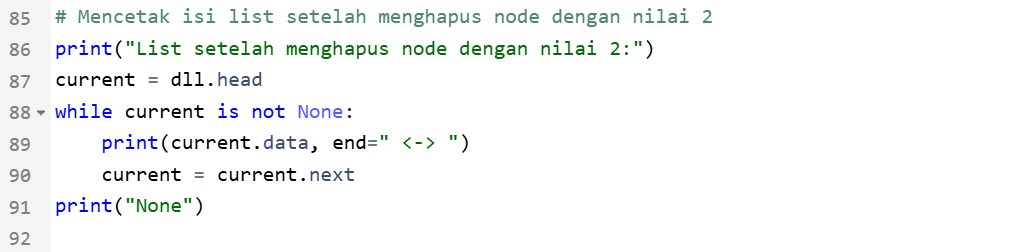
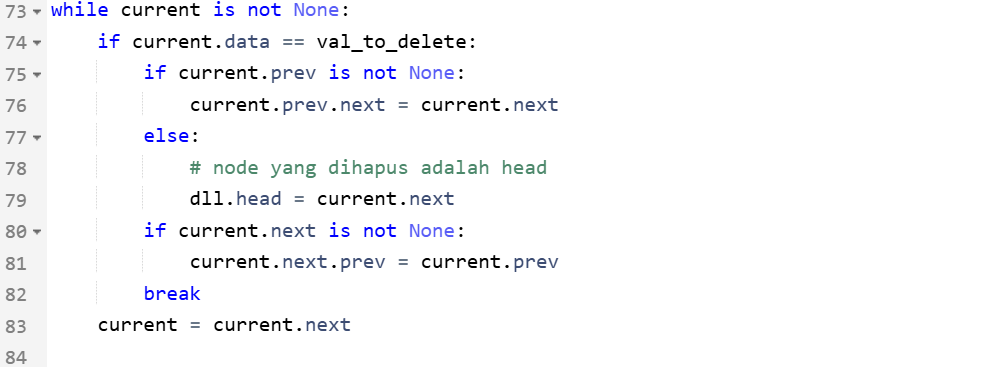
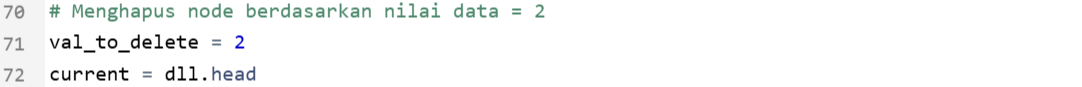
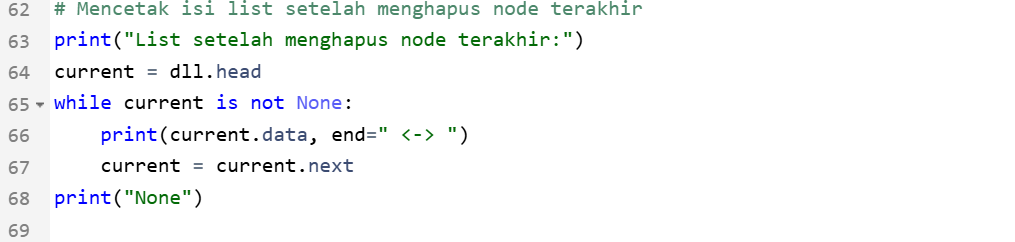
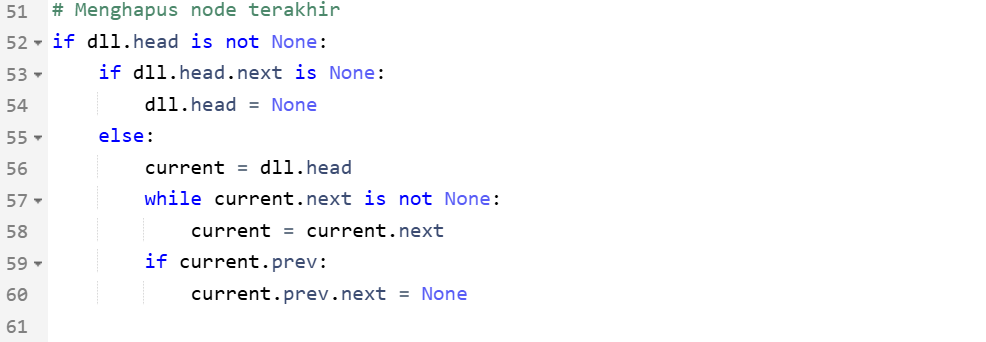
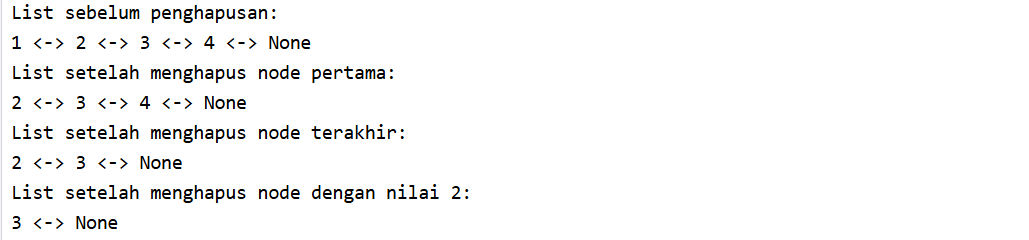
**Tugas penghapusan double linked list**





**Outputnya**

****

Penjelasannya

**Baris 1:** Mendefinisikan kelas Node sebagai struktur data dasar untuk elemen dalam double linked list.  
**Baris 2:** Mendefinisikan konstruktor \_\_init\_\_ untuk kelas Node.  
**Baris 3:** Menyimpan nilai yang diterima dalam self.data.  
**Baris 4:** Mengatur referensi prev dan next menjadi None.

**Baris 6:** Mendefinisikan kelas DoublyLinkedList sebagai wadah untuk menyimpan beberapa node.  
**Baris 7:** Mendefinisikan konstruktor \_\_init\_\_ untuk kelas tersebut.  
**Baris 8:** Inisialisasi self.head menjadi None, yang berarti list masih kosong.

**Baris 10:** Membuat objek dll dari kelas DoublyLinkedList.

**Baris 12:** Memulai perulangan untuk menambahkan data 1, 2, 3, 4 ke dalam linked list.  
**Baris 13**: Membuat objek new\_node dari kelas Node dengan nilai val.  
**Baris 14:** Mengecek apakah list masih kosong (dll.head is None).  
**Baris 15:** Jika list kosong, new\_node menjadi node pertama (head).  
**Baris 16:** Jika list tidak kosong, node akan ditambahkan di akhir.  
**Baris 17:** Inisialisasi current ke dll.head untuk mulai pencarian node terakhir.  
**Baris 18:** Perulangan untuk mencari node terakhir (current.next is not None).  
**Baris 19:** Berpindah ke node berikutnya.  
**Baris 20:** Menambahkan node baru di akhir, dan menghubungkannya dua arah.

**Baris 23:** Menampilkan teks "List sebelum penghapusan".  
**Baris 24**: Inisialisasi variabel current dengan dll.head.  
**Baris 25**: Melakukan perulangan untuk mencetak isi list.  
**Baris 26:** Mencetak nilai data node saat ini.  
**Baris 27:** Berpindah ke node berikutnya.  
**Baris 28:** Mencetak "None" sebagai penanda akhir list.

**Baris 31:** Mengecek apakah list tidak kosong sebelum menghapus node pertama.  
**Baris 32:** Jika hanya ada satu node, langsung hapus.  
**Baris 33:** Mengosongkan list jika hanya satu node.  
**Baris 34:** Jika lebih dari satu node, lanjut ke penghapusan.  
**Baris 35:** Geser head ke node kedua (dll.head.next).  
**Baris 36:** Set prev dari node baru menjadi None.

**Baris 39:** Menampilkan teks "List setelah menghapus node pertama".  
**Baris 40:** Inisialisasi ulang current ke dll.head.  
**Baris 41–43:** Loop untuk mencetak isi list.  
**Baris 44:** Mencetak "None" di akhir.

**Baris 46:** Mengecek apakah list tidak kosong sebelum menghapus node terakhir.  
**Baris 47:** Jika hanya ada satu node, langsung kosongkan list.  
**Baris 48:** Set dll.head = None.  
**Baris 49:** Jika lebih dari satu node, cari node terakhir.  
**Baris 50:** Inisialisasi current ke dll.head.  
**Baris 51:** Cari node terakhir dengan perulangan.  
**Baris 52:** Berpindah ke node berikutnya.  
**Baris 53:** Cek apakah ada node sebelumnya.  
**Baris 54**: Putuskan hubungan dengan node terakhir (current.prev.next = None).

**Baris 57:** Menampilkan teks "List setelah menghapus node terakhir".  
**Baris 58–61:** Melakukan pencetakan isi list.  
**Baris 62:** Cetak "None".

**Baris 64:** Menetapkan nilai yang ingin dihapus (val\_to\_delete = 2).  
**Baris 65:** Inisialisasi current ke dll.head.  
**Baris 66:** Mulai loop untuk mencari node dengan data 2.  
**Baris 67:** Jika data cocok...  
**Baris 68:** Jika node bukan head, sambungkan prev.next ke next.  
**Baris 69:** Hapus hubungan dari node sebelumnya ke node saat ini.  
**Baris 70:** Jika node adalah head, geser head ke node berikutnya.  
**Baris 71:** Lanjutkan proses penghapusan.  
**Baris 72:** Jika bukan node terakhir, sambungkan next.prev ke node sebelumnya.  
**Baris 73:** Menyambung node selanjutnya ke node sebelumnya.  
**Baris 74:** Keluar dari loop setelah penghapusan.  
**Baris 75:** Jika data belum cocok, lanjut ke node berikutnya.

**Baris 77:** Menampilkan teks "List setelah menghapus node dengan nilai 2".  
**Baris 78–81:** Melakukan pencetakan isi list.  
**Baris 82:** Cetak "None".