**LAPORAN TUGAS 9A**

**MATA KULIAH DATA STRUKTUR**

Dosen : Dwi Novia Prasetyanti,S.Kom, M.Cs.



**Dibuat Oleh :**

**Muhammad Hadist Rifannan (240202107)**

**Kelas TI-1D**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**POLITEKNIK NEGERI CILACAP**

**TAHUN AKADEMIK 2024/2025**

**Pendahuluan**

Struktur data merupakan salah satu aspek fundamental dalam ilmu komputer yang berperan penting dalam efisiensi pengolahan data dan pemrograman. Dengan pemahaman yang baik terhadap struktur data, seorang programmer dapat memilih metode penyimpanan dan manipulasi data yang paling sesuai untuk menyelesaikan suatu permasalahan secara optimal.

Dalam praktikum Struktur Data, mahasiswa diajak untuk mengimplementasikan berbagai jenis struktur data seperti array, linked list, stack, queue, tree, dan graph melalui bahasa pemrograman. Praktikum ini bertujuan agar mahasiswa tidak hanya memahami konsep teoritis, tetapi juga mampu mengaplikasikan konsep tersebut dalam bentuk kode program yang efisien dan fungsional.

Melalui laporan ini, akan dibahas secara rinci hasil praktikum yang mencakup tujuan percobaan, langkah-langkah pelaksanaan, hasil pengujian program, serta analisis terhadap kinerja dan keefektifan dari struktur data yang digunakan.

**Isi Laporan**

Membuat kelas Node untuk membuat blueprint dari tiap tiap node yang berisikan variable data , prev , next. Dan juga berisi fungsi konstruksi untuk membuat objek secara default pada kelas.

Jika objek node di inisialisasi ini akan memanggil fungsi construct secara otomatis sehingga akan terbuat Node baru di list tersebut .



class DoubleLinkedList {

Ini adalah deklarasi kelas bernama DoubleLinkedList.

Kelas ini akan berisi seluruh operasi dan struktur data untuk double linked list, seperti insert, delete, dan traverse.

private $head;

Deklarasi variabel private $head.

Variabel ini menyimpan referensi ke node pertama (paling depan) dalam list.

private artinya variabel ini hanya bisa diakses dari dalam kelas itu sendiri.

private $tail;

Deklarasi variabel untuk node terakhir (paling belakang) dari list.

Sama seperti $head, hanya dapat diakses dari dalam kelas DoubleLinkedList.

public function \_\_construct() { ... }

Ini adalah konstruktor: metode khusus yang akan dijalankan otomatis saat objek dibuat.

Konstruktor ini digunakan untuk menginisialisasi daftar agar kosong pada saat objek DoubleLinkedList pertama kali dibuat.

$this->head = null;

Menyatakan bahwa tidak ada node pertama saat list dibuat (kosong).

null berarti belum ada objek Node yang dihubungkan.

$this->tail = null;

Sama seperti di atas, belum ada node terakhir karena list masih kosong.



Tujuan:

Untuk menambahkan node baru di bagian depan (head) dari double linked list.

Langkah-Langkah:

1. Membuat node baru:

php

Salin

Edit

$newNode = new Node($data);

Membuat node baru berisi $data.

Node ini nantinya akan menjadi node paling depan di list.

2. Cek apakah list masih kosong:

php

Salin

Edit

if ($this->head === null)

Jika head bernilai null, berarti list kosong (belum ada node sama sekali).

3. Jika list kosong:

php

Salin

Edit

$this->head = $this->tail = $newNode;

Node baru menjadi satu-satunya node di list.

Karena list hanya punya 1 node, maka head dan tail menunjuk ke node tersebut.

4. Jika list tidak kosong:

php

Salin

Edit

$newNode->next = $this->head;

$this->head->prev = $newNode;

$this->head = $newNode;

Langkah ini untuk menyisipkan node baru di depan:

Sambungkan newNode.next ke node yang sebelumnya menjadi head.

Sambungkan head.prev ke newNode.

Perbarui head agar menunjuk ke newNode.

Hasil akhir:

Node baru kini menjadi node paling depan, dan terhubung dengan node lama sebagai next-nya. Node lama juga tahu bahwa node baru adalah prev-nya.

Kesimpulan:

Fungsi ini memungkinkan kita menambahkan data dari bagian depan double linked list dengan tetap menjaga koneksi antar node baik ke depan (next) maupun ke belakang (prev).

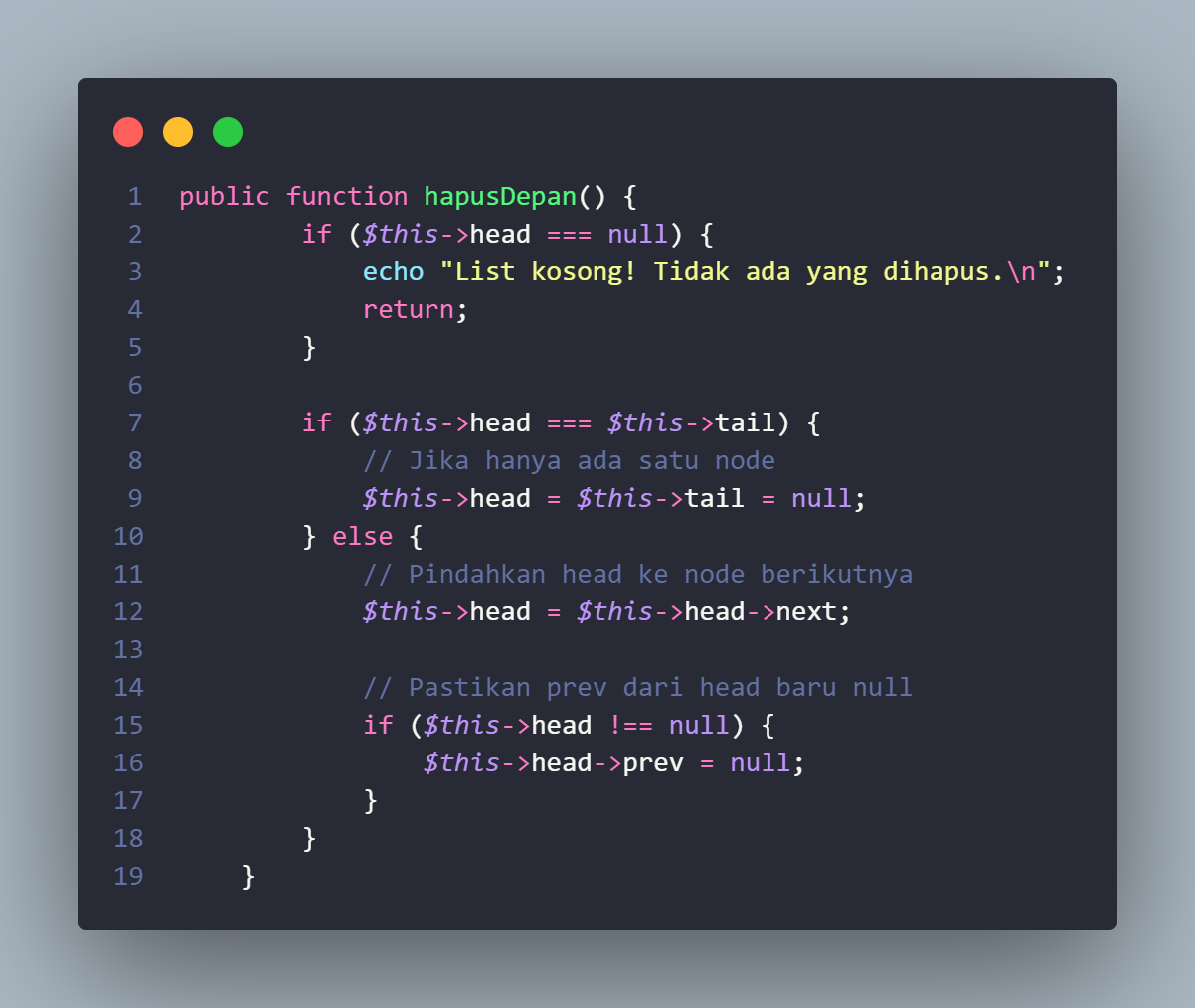


Fungsi insertBelakang digunakan untuk menambahkan node baru di bagian akhir dari struktur data double linked list. Prosesnya dimulai dengan membuat sebuah node baru yang berisi data yang ingin dimasukkan. Node ini kemudian akan diletakkan di belakang node terakhir saat ini (tail).

Pertama, fungsi akan memeriksa apakah linked list masih kosong, yaitu dengan melihat apakah nilai tail adalah null. Jika list kosong, maka node baru tersebut akan menjadi satu-satunya node di dalam list, sehingga head dan tail akan sama-sama menunjuk ke node yang baru saja dibuat.

Namun, jika list tidak kosong, maka node baru akan dihubungkan dengan node terakhir yang sudah ada. Langkah-langkahnya adalah sebagai berikut: pointer prev dari node baru dihubungkan ke node tail yang lama, lalu pointer next dari node tail lama dihubungkan ke node baru tersebut. Setelah hubungan dua arah terbentuk, pointer tail diperbarui untuk menunjuk ke node baru, menandakan bahwa node tersebut sekarang adalah node terakhir dalam list.

Dengan demikian, fungsi ini menjaga integritas double linked list dengan memastikan hubungan dua arah antara node-node tetap terjaga, dan node baru berhasil ditambahkan di bagian paling belakang list.



Fungsi hapusDepan() digunakan untuk menghapus node yang berada di bagian depan (paling awal) dari double linked list.

Langkah pertama dalam fungsi ini adalah memeriksa apakah list masih kosong, yaitu dengan melihat apakah head bernilai null. Jika list kosong, maka tidak ada node yang bisa dihapus, dan program akan mencetak pesan bahwa list kosong.

Jika list tidak kosong, fungsi kemudian memeriksa apakah head dan tail menunjuk ke node yang sama. Ini berarti hanya ada satu node dalam list. Jika benar, maka penghapusan node tersebut akan mengosongkan list sepenuhnya, sehingga head dan tail keduanya diatur menjadi null.

Namun, jika terdapat lebih dari satu node dalam list, maka proses selanjutnya adalah memindahkan pointer head ke node berikutnya, yaitu head->next. Setelah itu, agar hubungan antar node tetap benar, fungsi memastikan bahwa prev dari head yang baru di-set ke null, karena sekarang node ini adalah yang paling depan dan tidak boleh memiliki node sebelumnya.

Dengan kata lain, fungsi ini menghapus node pertama dan memastikan pointer-pointer dalam list diperbarui dengan benar agar struktur list tetap konsisten.

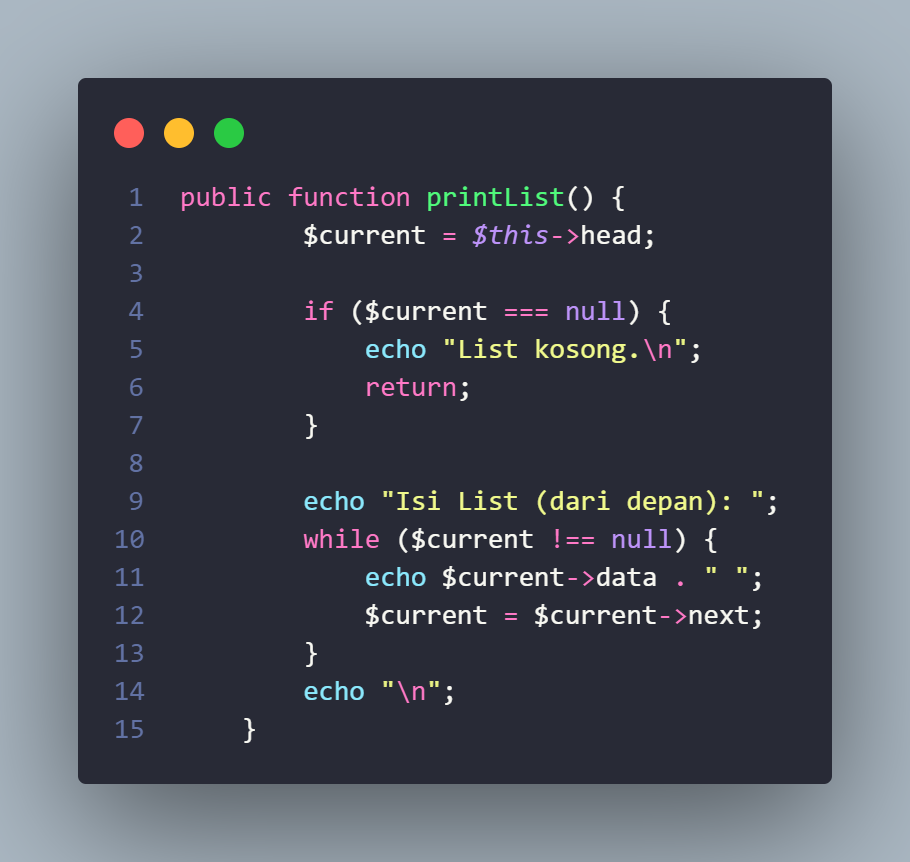


Fungsi hapusBelakang() berfungsi untuk menghapus node yang berada di bagian belakang (paling akhir) dari double linked list.

Pertama, fungsi akan memeriksa apakah list masih kosong dengan mengecek apakah tail bernilai null. Jika list kosong, maka tidak ada node yang bisa dihapus, dan program akan menampilkan pesan bahwa list kosong.

Jika list tidak kosong, langkah selanjutnya adalah memeriksa apakah head dan tail menunjuk ke node yang sama. Ini berarti hanya ada satu node dalam list. Jika itu terjadi, maka penghapusan node tersebut akan mengakibatkan list menjadi kosong, sehingga head dan tail diatur menjadi null.

Namun, jika terdapat lebih dari satu node dalam list, maka fungsi akan memindahkan tail ke node sebelumnya, yaitu tail->prev. Setelah itu, agar struktur list tetap benar, pointer next dari tail yang baru diatur menjadi null, karena node ini sekarang menjadi node paling akhir dan tidak boleh menunjuk ke node selanjutnya.



Fungsi printList() digunakan untuk menampilkan seluruh isi double linked list dari depan ke belakang.

Langkah pertama dalam fungsi ini adalah membuat variabel penunjuk bernama $current yang diisi dengan node head, yaitu node paling awal dalam list.

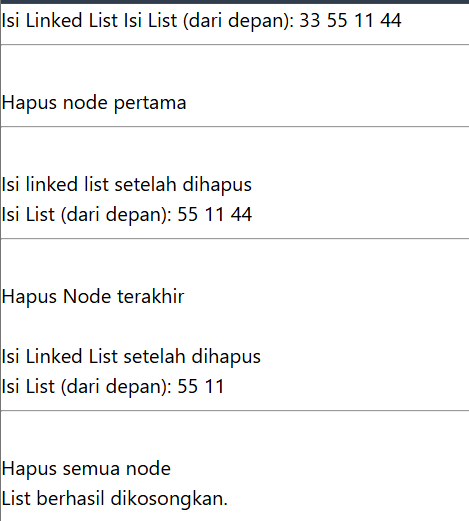
Selanjutnya, dilakukan pengecekan apakah $current bernilai null. Jika iya, artinya list masih kosong dan tidak ada data yang bisa ditampilkan, sehingga program akan mencetak pesan bahwa list kosong dan langsung keluar dari fungsi.

Jika list tidak kosong, maka program akan mencetak label atau keterangan awal bahwa isi list akan ditampilkan dari depan. Setelah itu, fungsi masuk ke dalam perulangan while yang akan terus berjalan selama $current tidak bernilai null.

Di dalam perulangan tersebut, setiap node akan diakses satu per satu mulai dari head. Data yang terdapat pada node saat ini dicetak ke layar, kemudian pointer $current digeser ke node berikutnya melalui $current->next.

Perulangan ini akan berhenti jika $current sudah mencapai null, yaitu setelah melewati node terakhir. Setelah selesai mencetak seluruh data, program mencetak baris baru sebagai penutup.

Secara keseluruhan, fungsi ini berfungsi sebagai alat bantu untuk melihat isi list secara urut dari depan ke belakang, dan sangat berguna untuk debugging atau verifikasi hasil operasi insert dan delete.



Diatas adalah output dari program yang dibuat

Fungsi clearList() digunakan untuk menghapus seluruh isi dari double linked list sekaligus. Caranya sangat sederhana, yaitu dengan mengatur pointer head dan tail menjadi null.

Dengan mengosongkan kedua pointer tersebut, maka tidak ada lagi node yang bisa diakses atau ditelusuri dalam list. Meskipun secara teknis objek node lama masih tersimpan di memori untuk sementara waktu, tapi mereka tidak lagi terhubung dalam list dan akan otomatis dihapus oleh garbage collector PHP jika tidak ada referensi lain.

Fungsi ini sangat berguna saat pengguna ingin memulai ulang list tanpa harus menghapus satu per satu dari depan atau belakang.

Sebagai tambahan, fungsi ini juga menampilkan pesan bahwa list berhasil dikosongkan sebagai feedback kepada pengguna.



