Quiz 2 Algoritma Sistem Data

TI 1D_Satria Wiguna_26

Genap

Soal No 1

```
//Soal No 1 Kode Genap
void deleteLast(){
    if (isEmpty()) {
        System.out.println(x:"linked list kosong, tidak dapat dihapus");
    } else if (head.n == null) {
        head = tail = null;
        size--;
    } else {
        Node temp = head;
        while (temp.n.n!= null) {
            temp = temp.n;
        }
        tail = temp;
        tail.n = null;
        size--;
```

Metode deleteLast() ini berfungsi untuk menghapus node terakhir dari daftar tertaut ganda (DoubleLinkedList).

Langkah-Langkah:

1. **Memeriksa Daftar Kosong:**
   - Jika daftar kosong (isEmpty() mengembalikan true), kode akan mencetak pesan yang menyatakan "linked list kosong, tidak dapat dihapus". Tidak ada perubahan yang dilakukan pada daftar.
2. **Memeriksa Daftar dengan Satu Node:**
   - Jika daftar hanya memiliki satu node (head.next == null), kode akan mengatur kedua pointer head dan tail menjadi null karena tidak ada lagi node tersisa. Ukuran daftar (size) dikurangi 1.
3. **Memproses Daftar dengan Banyak Node:**
   - Jika daftar memiliki lebih dari satu node:
   - Sebuah node sementara temp diinisialisasi untuk menunjuk ke head.
   - Loop akan berjalan hingga temp.next.next tidak sama dengan null. Kondisi ini memastikan kita berhenti sebelum node kedua dari belakang.
   - Di setiap iterasi, temp digerakkan ke depan untuk menunjuk ke node sebelum node terakhir.
   - Setelah loop selesai, tail diperbarui untuk menunjuk ke temp, yang secara efektif menjadi node kedua dari belakang yang baru.
   - Pointer next dari tail (yang awalnya menunjuk ke node terakhir) diatur menjadi null untuk memutuskan koneksi node terakhir dari daftar.

- Ukuran daftar (size) dikurangi 1 untuk mencerminkan penghapusan node

Soal no 2 :

```
//Soal No 2 Kode Ganjil dan Genap
void printFromTail() {
    Node temp = tail;
    while (temp!= null) {
        System.out.print("" + temp.data + "-");
        temp = temp.p;
    }
    System.out.println(x:"");
}
```

Metode printFromTail() ini berfungsi untuk mencetak data dari setiap node dalam daftar tertaut ganda (DoubleLinkedList), dimulai dari node terakhir (Tail) hingga node pertama (Head).

Langkah-Langkah:

1. **Inisialisasi Node Sementara:**
   - node sementara diinisialisasi untuk menunjuk ke node tail (ekor) dari daftar. tail menyimpan referensi ke node terakhir.
2. **Iterasi Melalui Daftar:**
   - Sebuah loop while dijalankan selama temp tidak sama dengan null. Artinya, loop akan terus berjalan selama ada node yang tersisa dalam daftar.
   - Di setiap iterasi loop:
     - Data dari node yang ditunjuk oleh temp dicetak ke konsol, diikuti dengan tanda hubung (-).
     - temp diperbarui untuk menunjuk ke node sebelumnya dalam daftar menggunakan pointer prev (disimpan dalam atribut p dari class Node). Ini berarti temp bergerak mundur (ke arah kepala) melalui daftar.
3. **Mencetak Baris Baru:**
   - Setelah loop selesai, kode mencetak baris baru (\n) untuk memulai baris baru setelah semua data dicetak.

Soal no 3 :

```java
//No. 3 Kode Genap
public static void split(DoubleLinkedList dll) {
    if (dll.isEmpty() || dll.head == dll.tail) {
    } else {
        DoubleLinkedList dll2 = new DoubleLinkedList();
        Node slow = dll.head;
        Node fast = dll.head;

        while (fast.n!= null && fast.n.n!= null) {
            slow = slow.n;
            fast = fast.n.n;
        }

        dll2.head = slow.n;
        dll2.head.p = null;
        slow.n = null;
        dll2.tail = dll.tail;
        dll2.size = dll.size - (dll.size / 2);
        dll.size = dll.size / 2;

        System.out.println(x:"List 1:");
        dll.print();
        System.out.println(x:"List 2:");
        dll2.print();
```

Fungsi `split` membagi daftar tertaut ganda `dll` menjadi dua daftar baru (List 1 dan List 2) dengan ukuran (kira-kira) sama.

- Pointer `slow` bergerak selangkah setiap iterasi, sementara `fast` bergerak dua langkah.
- Pembagian dilakukan di mana `slow` berhenti.
- List 1 tetap sebagai `dll` asli, dan List 2 dibuat sebagai daftar baru yang berisi setengah sisanya.
- Terakhir, kedua daftar dicetak.

Hasil :

```
Original List:
150-15-10-10-45-
150-15-10-10-
10-10-15-150-
List 1:
150-15-
List 2:
10-10-
```

Penjelasan keseluruhan :

- Kelas `DoubleLinkedList` memiliki atribut `head` (kepala), `tail` (ekor), dan `size` (ukuran) untuk melacak node dalam daftar.
- Metode `isEmpty()`, `addFirst()`, `deleteFirst()`, dan `print()` disediakan untuk pengecekan kosong, penambahan di depan, penghapusan dari depan, dan pencetakan elemen.
- Kode `Odd Lines` (baris ganjil) dan `Even Lines` (baris genap) menandakan implementasi alternatif untuk fungsi `addLast()` (tambah di belakang) dan `deleteLast()` (hapus dari belakang).
- Kelas `Main` menunjukkan contoh penggunaan `DoubleLinkedList` dengan menambahkan, menghapus, dan mencetak elemen.
- Fungsi `split()` (baris genap) membagi daftar menjadi dua dengan ukuran (kira-kira) sama. Metode `merge()` (belum diisi) seharusnya menggabungkan dua daftar.

Link github :

https://github.com/AuroraSauces/Praktikum-algoritma-dan-sistem-data/tree/main/Quiz%202