MAKALAH IMPLEMENTASI

STRUKTUR DATA LINKEDLIST

Dosen Pengampu:

Rizqi Putri Nourma Budiarti, S.T., M.T

Nama: Galeh Ariya Irwana

NIM: 3130021003

PRODI SISTEM INFORMASI

FAKULTAS EKONOMI BISNIS DAN TEKNOLOGI DIGITAL

UNIVERSITAS NAHDLATUL ULAMA' SURABAYA

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Linked List atau dikenal juga dengan sebutan senarai berantai adalah struktur data yang terdiri dari urutan record data, dimana setiap record memiliki field yang menyimpan alamat/referensi dari record selanjutnya. Element data yang dihubungkan dengan link pada Linked List disebut Node. Biasanya didalam LinkedList terdapat istilah Head, Tail, Next, Previous. Dalam setiap Bahasa pemrograman mungkin sudah ada atau sudah disediakan implementasi struktur data Linked List ini, namun kita juga perlu tahu cara untuk mengimplementasikannya secara manual, agar dapat memhami bagaimana cara kerja atau perilaku dari struktur data Linked List ini.

1.2 Rumusan Masalah

1.2.1 Bagaimana contoh implementasi Struktur Data LinkedList?

1.2.2 Bagaimana contoh implementasi Struktur Double LinkedList?

1.2.3 Bagaimana contoh implementasi Struktur Data Circular LinkedList?

1.2.4 Bagaimana contoh implementasi Struktur Data Multiple LinkedList?

1.3 Tujuan

1.3.1 Mengetahui cara kerja Struktur Data LinkedList

1.3.2 Mengetahui cara kerja Struktur Data Double LinkedList

1.3.3 Mengetahui cara kerja Struktur Data Circular LinkedList

1.3.4 Mengetahui cara kerja Struktur Data Multiple LinkedList

1.4 Manfaat

1.4.1 Memberikan pemahaman mengenai logika dan juga cara kerja dari Struktur Data LinkedList

1.4.2 Mampu mengimplementasikan Struktur Data LinkedList

BAB II
PEMBAHASAN

2.1 Implementasi Linked List

Untuk membuat sebuah Implentasi LinkedList atau SingleLinkedList kita akan membuat tiga class yaitu Node, LinkedListImpl, dan SingleListApp. Dimana class Node ini akan berfungsi sebagai tempat datanya, mulai dari mengambil data dan mengubah data, atau istilahnya bisa kita sebut setter dan getter pada class Node ini.Tidak lupa kita juga harus membuat constructor untuk memasukkan data ke dalam sebuah field atau attribute.

```
3   public class Node {
4       private String data;
5       private Node link;
6
```

2.1 Gambar Class

```
7
8       public Node() {
9           link = null;
10          data = null;
11      }
12
13      public Node(String data, Node link) {
14          this.data = data;
15          this.link = link;
16      }
```

2.2 Gambar Constructor

```
21
22      public void setData(String data) { this.data = data; }
25
26      public Node getLink() { return link; }
29
30      public void setLink(Node link) { this.link = link; }
33  }
34
```

2.3 Gambar Setter and Getter

Kemudian untuk melakukan operasi tambah data, remove data, dan yang lainnya kita bisa menggunakan class LinkedListImpl. Class inilah yang akan berfungsi sebagai tempat semua logic operasi dilakukan.

```java
public class LinkedListImpl {
    private Node start;
    private Node end;
    private int size;

    public LinkedListImpl() {
        start = null;
        end = null;
        size = 0;
    }
```

2.4 Contoh code class LinkedListImpl

Kita tahu sebelumnya dalam class ini akan berisi operasi untuk menyimpan semua logic jadi kita akan menambahkan beberapa function, yang pertama kita akan tambahkan function isEmpty dan juga getSize, dimana isEmpty ini akan mengecek apakah data dilinkedlist itu ada isinya atau tidak dan function ini akan mengembalikan nilai Boolean. Kemudian function getSize berfungsi sebagai pengambilan ukuran data yang ada dalam LinkedList ini.

```java
    public boolean isEmpty() { return start == null; }

    public int getSize() { return size; }
```

2.5 Gambar kode function isEmpty and getSize

Kemudian selanjutanya kita juga butuh sebuah function untuk menambah data dari posisi depan dan juga posisi akhir, jadi datanya yang kita masukkan nanti dapat diletakkan di bagian awal sendiri dan bisa juga di bagian paling akhir dalam sebuah node.

```java
    public void insertStart(String data) {
        Node node = new Node(data, link: null);
        size++;

        if(start == null) {
            start = node;
            end = start;
        }
    }
```

2.6 Gambar Code addFirst

```
46
47      public void insertToN(String data, int n) {
48          Node node = new Node(data, link: null);
49          Node start = this.start;
50          n = n-1;
51
52          for(var i = 1; i < size; i++) {
53              if(i == n) {
54                  Node temp = start.getLink();
55                  start.setLink(node);
56                  start.setLink(temp);
57                  break;
58              }
59              start = start.getLink();
60          }
61          size++;
62      }
```

2.7 Gambar Kode addLast

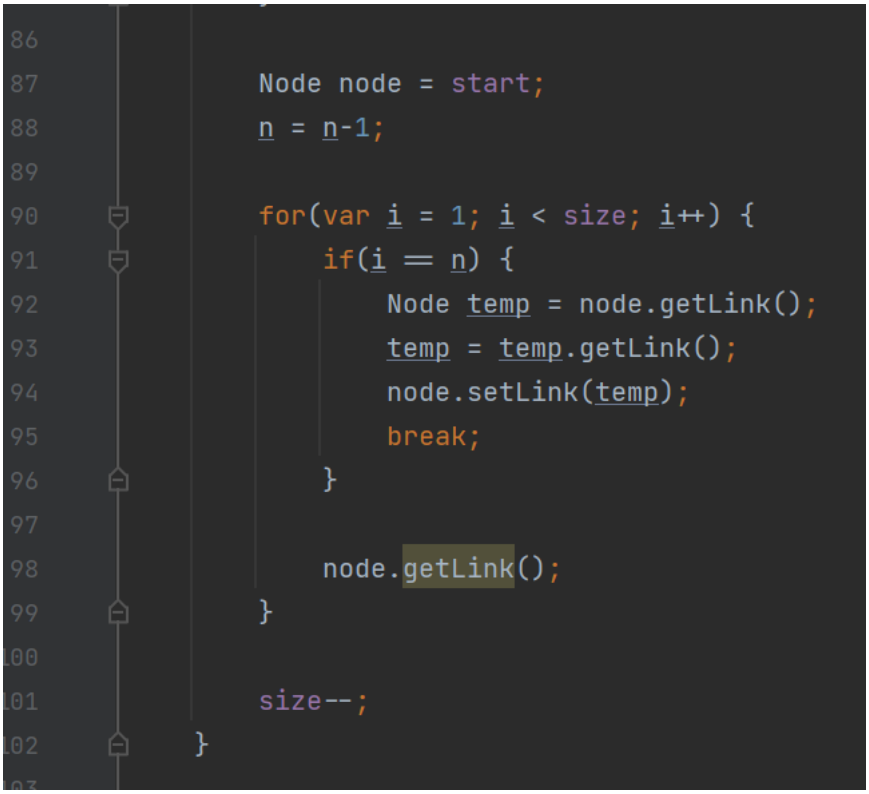
Selain dari insert atau memasukkan data dari awal ataupun dari akhir, tentu kita juga butuh memasukkan data di tengah-tengah atau datanya itu disisipkan. Kita akan membuat function insertToN dimana function ini akan menerima dua parameter yang pertama adalah datanya dan yang kedua adalah data tersebut mau dimasukan ke dalam urutan yang mana.

```
46
47      public void insertToN(String data, int n) {
48          Node node = new Node(data, link: null);
49          Node start = this.start;
50          n = n-1;
51
52          for(var i = 1; i < size; i++) {
53              if(i == n) {
54                  Node temp = start.getLink();
55                  start.setLink(node);
56                  start.setLink(temp);
57                  break;
58              }
59              start = start.getLink();
60          }
61          size++;
62      }
```

2.8 Gambar kode addAtN

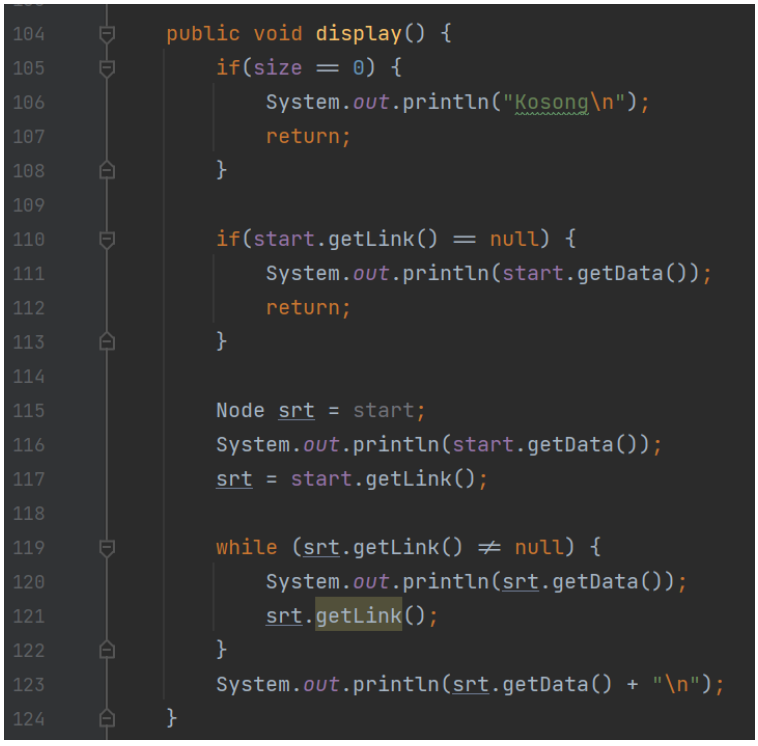
Setelah memberikan function menambahkan pada class ini, kita juga akan menambahkan dua function lagi yaitu remove dan juga display. Function remove akan digunakan untuk menghapus data yang ada dalam node dan fungsi hapus ini juga akan menerima sebuah parameter, dimana data dari parameter ini akan digunakan untuk menentukan urutan keberapa data yang akan dihapus. Kemudian juga function display, function display digunakan sebagai informasi dari data yang ada dalam node. Gambarannya apabila data nya kosong apa yang ditampilkan atau informasi yang lain mengenai data yang ada dalam node.

```
64
65      public void removeToN(int n) {
66          if(n == 1) {
67              start = start.getLink();
68              size--;
69              return;
70          }
71
72          if(n == size) {
73              Node s = start;
74              Node e = start;
75
76              while (s ≠ end) {
77                  e = s;
78                  s = s.getLink();
79              }
80
81              end = e;
82              end.setLink(null);
83              size--;
84              return;
85          }
```



```
86
87          Node node = start;
88          n = n-1;
89
90          for(var i = 1; i < size; i++) {
91              if(i == n) {
92                  Node temp = node.getLink();
93                  temp = temp.getLink();
94                  node.setLink(temp);
95                  break;
96              }
97
98              node.getLink();
99          }
100
101         size--;
102     }
```

2.9 Gambar kode removeToN



```
104     public void display() {
105         if(size == 0) {
106             System.out.println("Kosong\n");
107             return;
108         }
109
110         if(start.getLink() == null) {
111             System.out.println(start.getData());
112             return;
113         }
114
115         Node srt = start;
116         System.out.println(start.getData());
117         srt = start.getLink();
118
119         while (srt.getLink() != null) {
120             System.out.println(srt.getData());
121             srt.getLink();
122         }
123         System.out.println(srt.getData() + "\n");
124     }
```

2.10 Gambar kode function display

Sampai disni kita sudah membuat dua class yaitu Node sebagai representasi datanya dan juga LinkedListImpl sebagai business logic dari operasi yang akan kita lakukan terhadap datanya. dan yang terakhir kita butuh yang sebuah class lagi yaitu SingleListApp untuk menjalankan semua class yang akan kita buat, dalam istilah ini class ini disebut main class. Didalam class ini hanya akan ada satu method atau function yang bernama main.

```
import java.util.Scanner;

public class SingleListApp {
    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        LinkedListImpl list = new LinkedListImpl();
        System.out.println("Let's try Single List\n");

        char code;

        do {
            System.out.println("Perintah\n");
            System.out.println("1. Masukkan dari awal");
            System.out.println("2. Masukkan dari akhir");
            System.out.println("3. Masukkan dari ke N");
            System.out.println("4. Hapus posisi");
            System.out.println("5. Cek Kosong");
            System.out.println("6. Cek Size");

            System.out.print("\nMasukkan nomor:");
            int choice = scanner.nextInt();

            switch (choice) {
                case 1 -> {
                    System.out.println("Masukkan Data");
                    list.insertStart(scanner.nextLine());
                    break;
                }
                case 2 -> {
                    System.out.println("Masukkan Data");
                    list.insertEnd(scanner.nextLine());
                    break;
                }
                case 3 -> {
                    System.out.println("Masukkan Data");
                    String data = scanner.nextLine();
                    System.out.println("Enter Position");
                    int n = scanner.nextInt();

                    if(n < 1 || n > list.getSize()) {
                        System.out.println("Posisi tidak valid\n");
                    } else {
                        list.insertToN(data, n);
                        break;
                    }
                }
```

```
            case 4 -> {
                System.out.println("Masukkan Posisi");
                int p = scanner.nextInt();

                if(p < 1 || p > list.getSize()) {
                    System.out.println("Position tidak valid\n");
                } else  {
                    list.removeToN(p);
                    break;
                }
            }
            case 5 -> {
                System.out.println("Cek kosong: " + list.isEmpty());
                break;
            }
            case 6 -> {
                System.out.println("Size: " + list.getSize() + "\n");
                break;
            }
        }
        list.display();
        System.out.println("\nMau Lanjut? (type y or n)");
        code = scanner.next().charAt(0);

    } while (code == 'Y' || code == 'y');
```

2.11 Gambar kode Main