Praktikum Pemrograman 1

Tugas 5

Nama : Muhammad Faris Fathur Rohman

NRP : 223040126

Kelas : D

Link GitHub : https://github.com/Riss27/PP1_223040126_Pertemuan5.git

Node.java

```java
// Node.java
package Pertemuan_5;

public class Node {
    private int data;
    private Node next;

    /* inisialisasi atribut node */
    public Node(int data) {
        this.data = data;
    }

    /* getter Data */
    public int getData() {
        return data;
    }

    /* setter data */
    public void setData(int data) {
        this.data = data;
    }

    /* getter next */
    public Node getNext() {
        return next;
    }

    /* setter next */
    public void setNext(Node next) {
        this.next = next;
    }
}
```

Fungsi Remove Head, Mid, dan Tail

```java
// fungsi dispose
public void dispose(Node temp) {
    temp.setNext(null);
    temp = null;
}
```

Fungsi dispose digunakan untuk menghapus node yang tidak dibutuhkan lagi dalam linked list dengan mengatur pointer next dari node tersebut menjadi null dan menghapus referensi ke node tersebut.

```java
// algoritma fungsi removeHead
public void removeHead() {
    if (isEmpty()) {
        System.out.println("List kosong");
    } else {
        Node temp = HEAD;
        HEAD = HEAD.getNext();
        dispose(temp);
    }
}
```

Fungsi removeHead digunakan untuk menghapus node pertama dalam linked list. Jika linked list kosong, akan dicetak pesan "List kosong". Jika tidak, node pertama disimpan dalam variabel sementara, kemudian pointer HEAD digeser ke node berikutnya, dan memori yang dialokasikan untuk node yang dihapus dibebaskan.

```java
// algoritma fungsi removeMid
public void removeMid(int e) {
    Node preNode = new Node(0);
    Node tempNode = HEAD;
    boolean ketemu = false;
    int i = 1;

    if (isEmpty()) {
        System.out.println("Elemen list kosong");
    } else {
        while (tempNode != null && !ketemu) {
            if (tempNode.getData() == e) {
                ketemu = true;
            } else {
                preNode = tempNode;
                tempNode = tempNode.getNext();
                i++;
            }
        }

        if (ketemu) {
            if (i == 1) {
                HEAD = null;
            } else {
                preNode.setNext(tempNode.getNext());
                dispose(tempNode);
            }
        }
    }
}
```

Fungsi removeMid digunakan untuk menghapus node dengan nilai tertentu (e) dari linked list. Pencarian dilakukan dari awal hingga akhir linked list untuk menemukan node yang memiliki nilai yang sesuai. Jika ditemukan, node tersebut dihapus dari linked list dengan mengubah pointer dari node sebelumnya dan node setelahnya.

```java
// algoritma fungsi removeTail
public void removeTail() {
    Node preNode = null;
    Node lastNode;

    if (HEAD != null) {
        if (HEAD.getNext() == null) {
            HEAD = null;
        } else {
            lastNode = HEAD;
            while (lastNode.getNext() != null) {
                preNode = lastNode;
                lastNode = lastNode.getNext();
            }
            preNode.setNext(null);
            dispose(lastNode);
        }
    }
}
```

Fungsi removeTail digunakan untuk menghapus node terakhir dalam linked list. Jika linked list tidak kosong, pencarian dilakukan hingga node terakhir, kemudian node sebelumnya diatur untuk menunjuk ke null, dan memori yang dialokasikan untuk node terakhir dibebaskan.

Latihan dari modul Remove Head, Mid, dan Tail

```java
package Pertemuan_5;

// List (Operasi Remove Head & Tail)
// Latihan 2
public class StrukturListTest {
    public static void main(String[] args) {
        StrukturList list = new StrukturList(); // Create list

        // menambahkan elemen 2 di awal list
        list.addHead(2);
        // menambahkan elemen 9 di awal list
        list.addHead(9);
        // menambahkan elemen 7 di awal list
        list.addHead(7);

        // menampilkan elemen list
        System.out.println("Elemen list sebelum penghapusan:");
        list.displayElement();

        // menghapus elemen list di awal list
        list.removeHead();
        // menampilkan elemen list
        System.out.println("Elemen list setelah penghapusan:");
        list.displayElement();

        // menghapus elemen list di awal list 3x
        for (int i = 0; i < 3; i++) {
            list.removeHead();
            System.out.println("Elemen list setelah penghapusan ke-" + (i + 1) + ":");
            list.displayElement();
        }
    }
}
```

```java
package Pertemuan_5;
// List (Operasi Remove Head & Tail)
// Latihan 4
public class StrukturListTest1 {
    public static void main(String[] args) {
        StrukturList list = new StrukturList(); // Create list

        // menambahkan elemenn list
        list.addHead(1);
        list.addHead(5);
        list.addHead(3);
        list.addHead(6);
        list.addHead(2);

        // menampilkan elemen list
        System.out.println("Elemen list:");
        list.displayElement();

        // menghapus elemen di akhir list
        list.removeTail();
        // menampilkan elemen list
        System.out.println("Elemen list setelah penghapusan di akhir:");
        list.displayElement();

        // menghapus elemen di awal list
        list.removeHead();
        // menampilkan elemen list
        System.out.println("Elemen list setelah penghapusan di awal:");
        list.displayElement();
    }
}
```

```java
package Pertemuan_5;
// List (Operasi Remove Mid)
// Latihan 2
public class StrukturListTest2 {
    public static void main(String[] args) {
        // Create list
        StrukturList list = new StrukturList();

        // menambah elemen list berisi (2, 6, 3, 5, 1)
        list.addTail(2);
        list.addTail(6);
        list.addTail(3);
        list.addTail(5);
        list.addTail(1);

        // menampilkan elemen list
        System.out.println("Elemen List:");
        list.displayElement();

        // menghapus elemen 3 di tengah list
        list.removeMid(3);

        // menampilkan elemen list setelah penghapusan
        System.out.println("Elemen List setelah penghapusan:");
        list.displayElement();

        // Tambahan : menghapus elemen list ditengah
        list.removeMid(5);
        list.removeMid(6);

        // menampilkan elemen list setelah penghapusan tambahan
        System.out.println("Elemen List setelah penghapusan tambahan:");
        list.displayElement();
    }
}
```