Praktikum Pemrograman 1

Tugas 4

Nama : Muhammad Faris Fathur Rohman

NRP : 223040126

Kelas : D

Link GitHub : https://github.com/Riss27/Prakprog1_223040126.git

Latihan 1

```java
public class Node {
        private double data;
        private Node next;

        /* inisialisasi atribut node */
        public Node (double data2) {
                this.data = data2;
        }

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

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

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

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

Latihan 2

```
16          // algoritma fungsi addMid
17          public void addMid(double data, int position) { //int digantikan menjadi double
18                  Node newNode = new Node(data);
19                  Node curNode = null;
20                  Node posNode = null;
21                  int i;
22
23                  if(isEmpty()) {
24                          HEAD = newNode;
25                  }else {
26                          curNode = HEAD;
27                          if(position == 1) {
28                                  newNode.setNext(curNode);
29                                  HEAD = newNode;
30                          }else {
31                                  i = 1;
32                                  while((curNode != null)&&(i < position)) {
33                                          posNode = curNode;
34                                          curNode = curNode.getNext();
35                                          i++;
36                                  }
37                                  posNode.setNext(newNode);
38                                  newNode.setNext(curNode);
39                          }
40                  }
41          }
```

Metode ‘addHead’ berfungsi untuk menambahkan node baru di awal (head) dari linked list. Pertama-tama, sebuah objek baru, newNode dibuat dengan data yang diberikan. Selanjutnya, dilakukan pengecekan apakah linked list masih kosong atau tidak menggunakan metode isEmpty(). Jika linked list masih kosong, maka newNode akan menjadi HEAD langsung. Namun, jika linked list tidak kosong, newNode akan menjadi head baru, sementara node sebelumnya akan menjadi node setelah newNode.

```
16          // algoritma fungsi addMid
17          public void addMid(double data, int position) { //int digantikan menjadi double
18                  Node newNode = new Node(data);
19                  Node curNode = null;
20                  Node posNode = null;
21                  int i;
22
23                  if(isEmpty()) {
24                          HEAD = newNode;
25                  }else {
26                          curNode = HEAD;
27                          if(position == 1) {
28                                  newNode.setNext(curNode);
29                                  HEAD = newNode;
30                          }else {
31                                  i = 1;
32                                  while((curNode != null)&&(i < position)) {
33                                          posNode = curNode;
34                                          curNode = curNode.getNext();
35                                          i++;
36                                  }
37                                  posNode.setNext(newNode);
38                                  newNode.setNext(curNode);
39                          }
40                  }
41          }
```

Metode ‘addMid’ bertujuan untuk menambahkan node baru di tengah linked list pada posisi yang ditentukan. Prosesnya mirip dengan addHead, tetapi dengan tambahan iterasi hingga mencapai posisi yang diinginkan, sambil menyimpan node sebelum dan sesudah posisi yang dituju. Setelah mencapai posisi tersebut, newNode akan disisipkan di antara node sebelum dan sesudahnya.

```
43          // algoritma fungsi addTail
44          public void addTail(double data) { //int digantikan menjadi double
45                  Node newNode = new Node(data);
46
47                  if (isEmpty()) {
48                          HEAD = newNode;
49                  }else {
50                          Node posNode = null;
51                          Node curNode = HEAD;
52
53                          while(curNode != null) {
54                                  posNode = curNode;
55                                  curNode = curNode.getNext();
56                          }
57
58                          posNode.setNext(newNode);
59                  }
60          }
61          private boolean isEmpty() {
62                  return HEAD == null;
63          }
```

metode ‘addTail’ berperan dalam menambahkan node baru di akhir (ekor) dari linked list. Secara dasar, langkah-langkahnya mirip dengan addHead, tetapi iterasi dilakukan hingga mencapai node terakhir (node yang tidak memiliki node berikutnya). Kemudian, node terakhir tersebut dihubungkan dengan newNode, sehingga newNode menjadi node terakhir dalam linked list.