Modul 5 : ADT Double Linked List

5.1 Waktu Pelaksanaan Praktikum

Durasi kegiatan praktikum = **170 menit,** dengan rincian sebagai berikut (misalkan):

a. 15 menit untuk pengerjaan Tes Awal atau wawancara Tugas Pendahuluan
b. 60 menit untuk penyampaian materi
c. 45 menit untuk pengerjaan tugas / Studi Kasus
d. 50 menit **Pengayaan**

5.2 Tujuan

Setelah mengikuti praktikum ini mahasiswa diharapkan dapat:

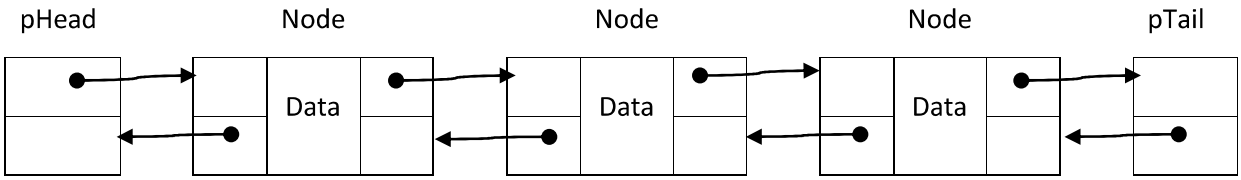
1. Mampu memahami konsep ADT double linked list
2. Mampu mengaplikasikan ADT double linked list

5.3 Alat & Bahan

1. Komputer
2. Java IDE

5.4 Dasar Teori

Double Linked List merupakan representasi data yang disimpan dalan suatu rangkaian node dimana setiap node mempunyai penunjuk ke node sebelumnya dan sesudahnya. Double : artinya field pointer-nya dua buah dan dua arah, yang menunjuk ke node sebelum dan sesudahnya. Berguna bila perlu melakukan pembacaan linkedlist dari dua arah. Double linked list memiliki 2 buah pointer yaitu pointer next dan prev. Pointer next : mengarah ke node belakang (tail). Pointer prev : mengarah ke node depan (head). Ilstrasi double Linked List dapat digambarkan berikut :



Ketika masih ada satu node maka kedua pointer (next dan prev) akan menunjuk ke NULL. Ketika double linked list memiliki banyak node maka node yang paling depan pointer prev-nya menunjuk ke NULL. Sedangkan node yang paling belakang pointer next-nya yang menunjuk ke NULL. Pada double linked dibutuhkan pointer bantu yaitu head dan tail. Head : menunjuk pada node yang paling depan. Tail : menunjuk pada node yang paling belakang. Dari ilustrasi gambar di atas ADT double Linked-list dapat direpresentasikan sebagai berikut :

Node
Data : Object
Next : Node
Prev : Node

DLL
head,tail: Node
size=0 : int
Inisialisasi():void
isEmpty(): boolean
size (): int
addFirst(Node input): void
addLast (Node input): void

5.5 Prosedur Praktikum

C. Percobaan Pertama: Memahami *Node* yang digunakan pada *Double Linked List*

1. Buatlah projek di IDE Java dengan nama DoubleLinkedListProject

2. Buatlah Class *Node* di projek *DoubleLinkedListProject* dan Tuliskan kode dibawah :

```
1   public class Node {
2       Object data;
3       Node next;
4       Node prev;
5       public static void main(String[] args)
6        {
7
8
9
10
11
12       }
13  }
```

3. Lakukan beberapa langkah berikut untuk memahami konsep Node pada Double Linked List:
   a. Tambahkan kode `Node head = new Node();` pada baris ke 7 class Node.
   b. Tambahkan kode `System.out.println("data : " + head.data);` pada baris ke 9 class Node kemudian jalankan program.
   c. Tambahkan kode `System.out.println("pointer: " + head.next);` pada baris ke 10 class Node kemudian jalankan program.
   d. Tambahkan kode `System.out.println("pointer: " + head.prev);` pada baris ke 11 class Node kemudian jalankan program.
   e. Tambahkan kode `head.data = "A";` pada baris ke 8 class Node kemudian jalankan program.
4. Tuliskan hasil percobaan, analisis hasil dan kesimpulannya.

D. Percobaan Kedua: Memahami Operasi sederhana pada Double Linked List

5. Tambahkan *Class DLL* di projek D*oubleLinkedListProject* dan Tuliskan kode dibawah :

```
1   public class DLL {
2     Node head,tail;
3     int size=0;
4
5     void inisialisasi(){
6       head=null;
7     }
8
9     boolean isEmpty(){
10       return (size==0);
11    }
12
13    int size()
14    {
15       return size;
16    }
17
18    void addFirst(Node input){
19      if (isEmpty()){
20          head=input;
21          tail=input;
```

```
22        }
23        else
24        {
25           input.next = head;
26           head.prev = input;
27           head = input;
28        } size++;
29     }
30
31     void addLast(Node input){
32        if (isEmpty()){
33           head = input;
34           tail = input;
35        }
36        else
37        {
38        input.prev = tail;
39        tail.next = input;
40        tail = input;
41        } size++;
42     }
43  }
```

6. Tambahkan method main pada class DLL dengan dengan kode sebagai berikut kemudian jalankan.

```
1   public static void main(String[] args)
2      {
3         DLL list = new DLL();
4         System.out.println("head : " + list.head);
5         System.out.println("tail : " + list.tail);
6         list.addFirst(new Node());
7         System.out.println("head : " + list.head);
8         System.out.println("tail : " + list.tail);
9         list.addFirst(new Node());
10        System.out.println("head : " + list.head);
11        System.out.println("tail : " + list.tail);
12        list.addLast(new Node());
13        System.out.println("head : " + list.head);
14        System.out.println("tail : " + list.tail);
15     }
```

7. Ganti isi dari method main pada class DLL dengan kode sebagai berikut kemudian jalankan.

```
1   public static void main(String[] args)
2      {
3         DLL list = new DLL();
4         System.out.println("head : " + list.head);
5         System.out.println("tail : " + list.tail);
6         list.addLast(new Node());
7         System.out.println("head : " + list.head);
8         System.out.println("tail : " + list.tail);
```

```
9         list.addLast(new Node());
10        System.out.println("head : " + list.head);
11        System.out.println("tail : " + list.tail);
12        list.addLast(new Node());
13        System.out.println("head : " + list.head);
14        System.out.println("tail : " + list.tail);
15    }
```

8. Tuliskan hasil percobaan, analisis hasil dan kesimpulannya.

5.6 Hasil Percobaan

1. Tuliskan hasil dari percobaan pertama diatas.
2. Tuliskan hasil dari percobaan kedua diatas.

5.7 Analisis Hasil

1. Tuliskan Analisis hasil dari percobaan pertama diatas.
2. Tuliskan Analisis hasil dari percobaan kedua diatas.

5.8 Kesimpulan

1. Tuliskan kesimpulan dari percobaan pertama diatas.
2. Tuliskan kesimpulan dari percobaan kedua diatas.

5.9 Latihan

Operasi pada Double Linked List secara lengkap adalah sebagai berikut:

1. Inisialisasi
2. isEmpty
3. size
4. Penambahan
5. Penghapusan
6. Penyisipan
7. Pencarian
8. Pengaksesan

Lengkapi kode class DLL diatas untuk operasi-operasi (method) yang belum ada pada Double Linked List.

5.10 Tugas

1. Modifikasilah program Latihan di atas sehingga DLL dapat menampung sembarang object. Kemudian, gunakan class Mahasiswa yang pernah dibuat pada modul 4 untuk diisikan pada DLL. Berikut class mahasiswa yang dimaksud:

Mahasiswa
String nim **String nama** **double ipk**
Constructor Mahasiswa **double getIpk** **String getNim** **String getNama**

2. Gunakan dan modifikasi method penyisipan data pada tugas di modul 4 yang bisa membentuk linked list urut sejak awal dengan pengurutan berdasarkan ipk.

3. Lengkapi method pada DLL yang bisa menampilkan data secara *ascending* dan *descending* berdasarkan ipk.

5.11 DAFTAR PUSTAKA

- Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, “Data Structures and Algorithms Using Java 6 edition”, Wiley, USA, 2014.
- John R. Hubbard, “Scaum’s Outline of Data Structures With Java second Edition”, McGraw-Hill, New york, 2007.
- Robert Lafore, “Data Structures and Algorithm in Java second Edition”, Sams Publishing, Indiana, 2003