LAPORAN HASIL PRAKTIKUM

Jobsheet 12

NAMA : Handino Asa Galih r

NIM : 244107020237

KELAS : 1-E

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLINEMA

2025

Praktikum 2.

Percobaan 1

Class Mahasiswa12.java

```
public class Mahasiswa12 {

    public String nim;

    public String nama;

    public String kelas;

    public double ipk;


    public Mahasiswa12(String nim, String nama, String kelas, double
ipk) {

        this.nim = nim;

        this.nama = nama;

        this.kelas = kelas;

        this.ipk = ipk;

    }


    public void tampil() {

        System.out.println("NIM: " + nim + ", Nama: " + nama + ", Kelas:
" + kelas + ", IPK: " + ipk);

    }

}
```

Class Node12.java

```
public class Node12 {
    Mahasiswa12 data;
    Node12 prev;
    Node12 next;

    public Node12(Mahasiswa12 data) {
        this.data = data;
        this.prev = null;
        this.next = null;
    }
}
```

Class DoubleLinkedList12.java

```
public class DoubleLinkedLists12 {
    Node12 head;
    Node12 tail;

    public DoubleLinkedLists12() {
        head = null;
        tail = null;
    }

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

```
    }

    public void addFirst(Mahasiswa12 data) {
        Node12 newNode = new Node12(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            newNode.next = head;
            head.prev = newNode;
            head = newNode;
        }
    }

    public void insertAfter(String keyNim, Mahasiswa12 data)
{
        Node12 current = head;
        while (current != null &&
!current.data.nim.equals(keyNim)
            current = current.next;
        }

        if (current == null) {
            System.out.println("Node dengan NIM " + keyNim +
" tidak ditemukan.");
            return;
        }

        Node12 newNode = new Node12(data);
        if (current == tail) {
```

```
            current.next = newNode;
            newNode.prev = current;
            tail = newNode;
        } else {
            newNode.next = current.next;
            newNode.prev = current;
            current.next.prev = newNode;
            current.next = newNode;
        }

        System.out.println("Node berhasil disisipkan setelah
NIM " + keyNim);
    }
    public void print() {
        if (isEmpty()) {
            System.out.println("List masih kosong.");
            return;
        }
        Node12 current = head;
        while (current != null) {
            current.data.tampil();
            current = current.next;
        }
    }

    public void removeFirst() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa
dihapus.");
```

```
        }
        if (head == tail) {
            head = tail = null;
        } else {
            head = head.next;
            head.prev = null;
        }
    }

    public void removeLast() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa
dihapus.");
            return;
        }
        if (head == tail) {
            head = tail = null;
        } else {
            tail = tail.prev;
            tail.next = null;
        }
    }
    }

    public Node12 search(String nim) {
        Node12 current = head;
        while (current != null) {
            if (current.data.nim.equals(nim)) {
                return current;
```

```
            }
            current = current.next;
        }
        return null;
    }
}
```

Class DLLMain

```
import java.util.Scanner;


public class DLLmain {
    public static void main(String[] args) {
        DoubleLinkedLists12 list = new
DoubleLinkedLists12();
        Scanner scan = new Scanner(System.in);
        int pilihan;


        do {
            System.out.println("\nMenu Double Linked List
Mahasiswa");
            System.out.println("1. Tambah di awal");
            System.out.println("2. Tambah di akhir");
            System.out.println("7. Hapus di awal");
            System.out.println("8. Hapus di akhir");
            System.out.println("9. Tampilkan data");
            System.out.println("13. Cari Mahasiswa
berdasarkan NIM");
            System.out.println("0. Keluar");
```

```
            System.out.print("Pilih menu: ");
            pilihan = scan.nextInt();
            scan.nextLine();

            switch (pilihan) {
                case 1 -> {
                    Mahasiswa12 mhs = inputMahasiswa(scan);
                    list.addFirst(mhs);
                }
                case 2 -> {
                    Mahasiswa12 mhs = inputMahasiswa(scan);
                    list.addLast(mhs);
                }
                case 3 -> {
                   list.removeFirst();
                    break;
                }
                case 4 -> {
                    List.removeLast();
                    Break;
                }
                case 5 -> {
                    list.print();
                    break;
                }
                case 6 -> {
                    System.out.print("Masukkan indeks data
yang mau dihapus: ");
```

```
                case 6 -> {
                    System.out.print("Masukkan NIM yang
ingin dicari: ");
                    String nimCari = sc.nextLine();
                    list.remove(index);
                }
                case 7 -> {
                    System.out.print("Masukkan NIM yang
dicari: ");
                    String nim = scan.nextLine();
                    Node12 found = list.search(nimCari);
                    if (found != null) {
                        System.out.println("Data
ditemukan:");
                        found.data.tampil();
                    } else {
                        System.out.println("Data tidak
ditemukan.");
                    }
                case 0 -> System.out.println("Keluar dari
program.");
                default -> System.out.println("Pilihan tidak
valid.");
            }
        } while (pilihan != 0);
        sc.close();
    }
}
```

Output

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 1
Masukkan NIM: 20304050
Masukkan Nama: Hermione
Masukkan Kelas: Gryffindor
Masukkan IPK: 4

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 5
NIM: 20304050, Nama: Hermione, Kelas: Gryffindor, IPK: 4.0
```

Pertanyaan

1. Jelaskan perbedaan antara single linked lisr dengan double linked list!
   Single Linked List :
   - Hanya memiliki satu pointer di setiap node yang akan merujuk ke node berikutnya.
   - Menggunakan ruang memori lebih sedikit karena hanya satu pointer setiap node.
   - Harus dimulai dari head untuk mencapai node sebelumnya.
   - Traversal hanya digunakn dalam satu arah

   Double Linked List :

   - Memiliki dua pointer di setiap node : next (digunakan untuk menunjuk ke node berikutnya) dan prev (digunakan untuk menunjuk ke node sebelumnya).
   - Traversal dapat dilakukan dengan dua arah.
   - Node berikutnya dapt diakses secara langsung.
   - Menggunakan ruang memori lebih banyak karena per node memilki dua pointer.

2. Perhatikan class Node01, di dalamnya terdapat atribut next dan prev. Untuk apakah atribut tersebut?

   Atribut Next :
   - Dalam linked list digunkan untuk menyimpan referensi Alamat ke node berikutnya.
   - Memungkinkan traversal dari node saat ini ke node yang ada setelahnya.
   - Jika Node sudah mencapai Node terakhir, maka Ketika Next akan bernilai null.

   Atribut Prev :

   - Sama sama digunakan untuk menyimpan referensi Alamat ke Node berikutnya.
   - Memungkinkan traversal mundur dari node saat ini ke node yang sebelumnya.
   - Jika node paling awal, maka prev akan bernilai null.

3. Perhatikan kosntruktor pada class DoubleLinkedList. Apa kegunaan dari kosntruktor tersebut?

```
public DoubleLinkedList01() {
    head = null;
    tail = null;
}
```

   Fungsi konstruktor DoubleLunkedList() digunakan untuk :
   - Ketika double linked list dalam keadaan kosong maka akan diinisialisasi
   - Mengatur head dan tail menjadi null, yang menandakan belum ada node dalam list.

4. Pada node addFirst(), apa yang dimaksud dari kode berikut?

```
if (isEmpty()) {
    head = tail = newNode;
```

   If(isEmpty) : mengecek list apakah masih kosong
   Jika lsit masih kosong head dan tail akan diatur ke newNode, karena node baru adalah satu satunya node.

5. Perhatikan pada method addFirst(). Apakah arti statement head.prev = newNode ?

   Statement head.prev = newNode yang berarti :
   - Mengatur dari node head ke pointer prev (yang lama) yang menunjuk ke node baru.
   - Menghubungkan koneksi dua arah antara node baru dengan head lama.
   - Memastikan head lama mengakses node baru sebagai node sebelumnya.

6. Modifikasi code pada fungsi print() agar dapat menampilkan warning atau pesan bahwa linked list masih dalam kondisi.

```
    public void print() {
        if (isEmpty()) {
            System.out.println(x:"List masih kosong.");
            return;
        }
```

7. Pada insertAfter(), apa yang dimaksud dari kode berikut ?

```
current.next.prev = newNode;
```

current.next : digunkan untuk mengakses node yang ada setelah current.
.prev = newNode : untuk mengatur pointer prev dari node setelah current menunjuk ke newNode.
Yang berfungsi untuk menghubungkan node lama untuk disisipkan ke node baru.

8. Modifikasi menu pilihan dan switch-case agar fungsi insertAfter() masuk ke dalam menu pilihan dan dapat berjalan dengan baik.

Class DLLmain :

```
case 13 -> {
    System.out.print(s:"Masukkan NIM yang dicari: ");
    String nim = scan.nextLine();
    Node12 found = list.search(nim);
    if (found != null) {
        System.out.println(x:"Data ditemukan:");
        found.data.tampil();
    } else {
        System.out.println(x:"Data tidak ditemukan.");
    }
```

Percobaan 2

Method removeFirst() dan removeLast() di dalam class DoubleLinkedList.

```
    public void removeFirst() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus.");
            return;
        }
        if (head == tail) {
            System.out.println("Data sudah berhasil dihapus. Data yang
terhapus adalah : ");
            head.data.tampil();
            head = tail = null;
        } else {
            System.out.println("Data sudah berhasil dihapus. Data yang
terhapus adalah : ");
            head.data.tampil();
            head = head.next;
            head.prev = null;
        }

        size--;
    }

    public void removeLast() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus.");
            return;
        }
        if (head == tail) {
            System.out.println("Data sudah berhasil dihapus. Data yang
terhapus adalah : ");
            tail.data.tampil();
            head = tail = null;
        } else {
            System.out.println("Data sudah berhasil dihapus. Data yang
terhapus adalah : ");
```

```
        } else {
            System.out.println("Data sudah berhasil dihapus. Data yang
terhapus adalah : ");
            tail.data.tampil();
            tail = tail.prev;
            tail.next = null;
        }

        size--;
    }
```

Output

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Masukkan data setelah NIM tertentu
0. Keluar
Pilih menu: 2
Masukkan NIM: 20304050
Masukkan Nama: Hermione
Masukkan Kelas: Gryffindor
Masukkan IPK: 4

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus dari awal
4. Hapus dari akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
7. Masukkan data setelah NIM tertentu
0. Keluar
Pilih menu: 3
```

Pertanyaan

1. Apakah maksud statement berikut pada method removeFirst()?

```
head = head.next;
head.prev = null;
```

- Head = head.next : Memindahkan pointer head ke node berikutnya, sehingga node pertama lama tidak lagi menjadi head.
- head.prev = null : Mengatur pointer prev dari head yang baru menjadi null, karena head tidak memiliki node sebelumnya.

2. Modifikasi kode program untuk menampilkan pesan “Data sudah berhasil dihapus. Data yang terhapus adalah … “

```
public void removeFirst() {
    if (isEmpty()) {
        System.out.println(x:"List kosong, tidak bisa dihapus.");
        return;
    }
    if (head == tail) {
        System.out.println(x:"Data sudah berhasil dihapus. Data yang terhapus adalah : ");
        head.data.tampil();
        head = tail = null;
    } else {
        System.out.println(x:"Data sudah berhasil dihapus. Data yang terhapus adalah : ");
        head.data.tampil();
        head = head.next;
        head.prev = null;
    }

    size--;
}

public void removeLast() {
    if (isEmpty()) {
        System.out.println(x:"List kosong, tidak bisa dihapus.");
        return;
    }
    if (head == tail) {
        System.out.println(x:"Data sudah berhasil dihapus. Data yang terhapus adalah : ");
        tail.data.tampil();
        head = tail = null;
    } else {
        System.out.println(x:"Data sudah berhasil dihapus. Data yang terhapus adalah : ");
        tail.data.tampil();
        tail = tail.prev;
        tail.next = null;
    }
```

Output

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 1
Masukkan NIM: 2441070
Masukkan Nama: Handino asa
Masukkan Kelas: 1E
Masukkan IPK: 3

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 2
Masukkan NIM: 123456
Masukkan Nama: Asa Galih R
Masukkan Kelas: 1E
Masukkan IPK: 4
```

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 3
Data berhasil dihapus. Data yang terhapus adalah Handino asa

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
6. Cari Mahasiswa berdasarkan NIM
0. Keluar
Pilih menu: 4
Data sudah berhasil dihapus. Data yang terhapus adalah Asa Galih R
```

Tugas

Penambahan fungsi Add(), removeAfter(), remove(), getfirst(), getlast(), getindex(), getSize()

```
public void add(int index, Mahasiswa12 data) {
        if (index < 0) {
            System.out.println("Indeks tidak valid.");
            return;
        }
        if (index == 0) {
            addFirst(data);
            return;
        }
        if (index >= size) {
            addLast(data);
            return;
        }

        Node12 current = head;
        int currentIndex = 0;
        while (current != null && currentIndex < index) {
            current = current.next;
            currentIndex++;
        }

        Node12 newNode = new Node12(data);

        newNode.prev = current.prev;
        newNode.next = current;

        if (current.prev != null) {
            current.prev.next = newNode;
        }
        current.prev = newNode;
```

```
        size++;
        System.out.println("Data berhasil ditambahkan di indeks " +
index);
    }
      public void print() {
        if (isEmpty()) {
            System.out.println("List masih kosong.");
            return;
        }
        Node12 current = head;
        while (current != null) {
            current.data.tampil();
            current = current.next;
        }
    }

      public void removeFirst() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus.");
            return;
        }
        if (head == tail) {
            System.out.println("Data sudah berhasil dihapus. Data yang
terhapus adalah : ");
            head.data.tampil();
            head = tail = null;
        } else {
            System.out.println("Data sudah berhasil dihapus. Data yang
terhapus adalah : ");
            head.data.tampil();
            head = head.next;
            head.prev = null;
        }

        size--;
```

```
        size--;
    }

    public void removeLast() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus.");
return;
        }
        if (head == tail) {
            System.out.println("Data sudah berhasil dihapus. Data yang
terhapus adalah : ");
            tail.data.tampil();
            head = tail = null;
        } else {
            System.out.println("Data sudah berhasil dihapus. Data yang
terhapus adalah : ");
            tail.data.tampil();
            tail = tail.prev;
            tail.next = null;
        }

        size--;
    }

    public void removeAfter(String keyNim) {
        Node12 current = head;

        while (current != null && !current.data.nim.equals(keyNim)) {
            current = current.next;
        }

        if (current == null) {
            System.out.println("Node dengan NIM " + keyNim + " tidak
ditemukan.");
        } else if (current.next == null) {
```

```
            System.out.println("Tidak ada node setelah NIM " + keyNim +
" yang bisa dihapus.");
        } else {
            Node12 toDelete = current.next;
            System.out.println("Data setelah " + keyNim + " berhasil
dihapus:");
            toDelete.data.tampil();

            current.next = toDelete.next;
            if (toDelete.next != null) {
                toDelete.next.prev = current;
     } else {
                tail = current;
            }
            toDelete.next = toDelete.prev = null;
            size--;
        }
    }

    public void remove(int index) {
        if (index < 0 || isEmpty()) {
            System.out.println("Indeks tidak valid atau list kosong.");
            return;
        }

        if (index == 0) {
            removeFirst();
            return;
        }

        Node12 current = head;
        int currentIndex = 0;

        while (current != null && currentIndex < index) {
            current = current.next;
            currentIndex++;
```

```
        }

        if (current == null) {
            System.out.println("Indeks di luar batas.");
        } else {
            System.out.println("Data pada indeks " + index + " berhasil
dihapus:");
            current.data.tampil();

            if (current == tail) {
                removeLast();

} else {
                current.prev.next = current.next;
                current.next.prev = current.prev;

                current.next = current.prev = null;
                size--;
            }
        }
    }

    public Node12 search(String nim) {
        Node12 current = head;
        while (current != null) {
            if (current.data.nim.equals(nim)) {
                return current;
            }
            current = current.next;
        }
        return null;
    }

    public void getFirst() {
        if (isEmpty()) {
```

```
            System.out.println("List kosong.");
        } else {
            System.out.println("Data pada node pertama:");
            head.data.tampil();
        }
    }

    public void getLast() {
        if (isEmpty()) {
            System.out.println("List kosong.");
        } else {
            System.out.println("Data pada node terakhir:")
            tail.data.tampil();
        }
    }

    public void getIndex(int index) {
        if (isEmpty()) {
            System.out.println("List kosong.");
            return;
        }

        Node12 current = head;
        int currentIndex = 0;

        while (current != null && currentIndex < index) {
            current = current.next;
            currentIndex++;
        }

        if (current == null) {
            System.out.println("Indeks di luar batas.");
        } else {
            System.out.println("Data pada indeks ke-" + index + ":");
            current.data.tampil();
```

```
            }
        }

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

Output

Add()

```
11. Tampilkan data terakhir
12. Tampilkan data berdasarkan indeks
13. Cari Mahasiswa berdasarkan NIM
14. Tampilkan jumlah data
0. Keluar
Pilih menu: 4
Masukkan indeks untuk menambahkan data: 2
Masukkan NIM: 091104
Masukkan Nama: Doni akbar
Masukkan Kelas: 1E
Masukkan IPK: 4
Data berhasil ditambahkan di indeks 2

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Tambah setelah NIM tertentu
4. Tambah pada indeks tertentu
5. Hapus setelah NIM tertentu
6. Hapus berdasarkan indeks
7. Hapus di awal
8. Hapus di akhir
9. Tampilkan data
10. Tampilkan data pertama
11. Tampilkan data terakhir
12. Tampilkan data berdasarkan indeks
13. Cari Mahasiswa berdasarkan NIM
14. Tampilkan jumlah data
0. Keluar
Pilih menu: 9
NIM: 232474, Nama: Rottweiler, Kelas: 1E, IPK: 3.0
NIM: 2441070, Nama: Handino Asa, Kelas: 1E, IPK: 4.0
NIM: 091104, Nama: Doni akbar, Kelas: 1E, IPK: 4.0
NIM: 123456, Nama: Bobby, Kelas: 1E, IPK: 2.0
NIM: 234567, Nama: Luna, Kelas: 1E, IPK: 3.0
```

removeAfter()

```
7. Hapus di awal
8. Hapus di akhir
9. Tampilkan data
10. Tampilkan data pertama
11. Tampilkan data terakhir
12. Tampilkan data berdasarkan indeks
13. Cari Mahasiswa berdasarkan NIM
14. Tampilkan jumlah data
0. Keluar
Pilih menu: 5
Masukkan NIM yang jadi acuan: 232474
Data setelah 232474 berhasil dihapus:
NIM: 2441070, Nama: Handino Asa, Kelas: 1E, IPK: 4.0

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Tambah setelah NIM tertentu
4. Tambah pada indeks tertentu
5. Hapus setelah NIM tertentu
6. Hapus berdasarkan indeks
7. Hapus di awal
8. Hapus di akhir
9. Tampilkan data
10. Tampilkan data pertama
11. Tampilkan data terakhir
12. Tampilkan data berdasarkan indeks
13. Cari Mahasiswa berdasarkan NIM
14. Tampilkan jumlah data
0. Keluar
Pilih menu: 9
NIM: 232474, Nama: Rottweiler, Kelas: 1E, IPK: 3.0
NIM: 091104, Nama: Doni akbar, Kelas: 1E, IPK: 4.0
NIM: 123456, Nama: Bobby, Kelas: 1E, IPK: 2.0
NIM: 234567, Nama: Luna, Kelas: 1E, IPK: 3.0
```

fungsiRemove()

```
6. Hapus berdasarkan indeks
7. Hapus di awal
8. Hapus di akhir
9. Tampilkan data
10. Tampilkan data pertama
11. Tampilkan data terakhir
12. Tampilkan data berdasarkan indeks
13. Cari Mahasiswa berdasarkan NIM
14. Tampilkan jumlah data
0. Keluar
Pilih menu: 6
Masukkan indeks data yang mau dihapus: 0
Data sudah berhasil dihapus. Data yang terhapus adalah :
NIM: 232474, Nama: Rottweiler, Kelas: 1E, IPK: 3.0

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Tambah setelah NIM tertentu
4. Tambah pada indeks tertentu
5. Hapus setelah NIM tertentu
6. Hapus berdasarkan indeks
7. Hapus di awal
8. Hapus di akhir
9. Tampilkan data
10. Tampilkan data pertama
11. Tampilkan data terakhir
12. Tampilkan data berdasarkan indeks
13. Cari Mahasiswa berdasarkan NIM
14. Tampilkan jumlah data
0. Keluar
Pilih menu: 9
NIM: 091104, Nama: Doni akbar, Kelas: 1E, IPK: 4.0
NIM: 123456, Nama: Bobby, Kelas: 1E, IPK: 2.0
NIM: 234567, Nama: Luna, Kelas: 1E, IPK: 3.0
```

getSize()

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Tambah setelah NIM tertentu
4. Tambah pada indeks tertentu
5. Hapus setelah NIM tertentu
6. Hapus berdasarkan indeks
7. Hapus di awal
8. Hapus di akhir
9. Tampilkan data
10. Tampilkan data pertama
11. Tampilkan data terakhir
12. Tampilkan data berdasarkan indeks
13. Cari Mahasiswa berdasarkan NIM
14. Tampilkan jumlah data
0. Keluar
Pilih menu: 14
Jumlah data: 3
```