JOBSHEET 12

ALGORITMA STRUKTUR DATA

Burhnauddin ihsan

244107020189

TI 1E/06

PROGRAM STUDI D-IV TEKNIK INFORMATIKA JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2025

PERCOBAAN 1

1. Kode program

   Mahasiswa06

```
public class Mahasiswa06 {

        String nim, nama, kelas;
        double ipk;

        Mahasiswa06() {
        }

        Mahasiswa06(String nim, String nama, String kelas, double ipk)
{
            this.nim = nim;
            this.nama = nama;
            this.kelas = kelas;
            this.ipk = ipk;
        }

        void tampilInformasi() {
            System.out.println("Nama : " + this.nama);
            System.out.println("NIM : " + this.nim);
            System.out.println("Kelas : " + this.kelas);
            System.out.println("IPK : " + this.ipk);
            System.out.println("====================================");
        }
}
```

   NodeMahasiswa06

```
public class Node06 {
    Mahasiswa06 data;
    Node06 prev;
    Node06 next;

   public Node06(Mahasiswa06 data) {
    this.data = data;
    this.prev = null;
    this.next = null;
   }

}
```

DoubleLinkedList

```
public class DoubleLinkedList06 {
    Node06 head;
    Node06 tail;

    public DoubleLinkedList06() {
        head = null;
        tail = null;
    }

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

    public void addFirst(Mahasiswa06 data) {
        Node06 newNode = new Node06(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            newNode.next = head;
            head.prev = newNode;
            head = newNode;
        }
    }

    public void addLast(Mahasiswa06 data) {
        Node06 newNode = new Node06(data);
        if (isEmpty()) {
            head = tail = newNode;
        } else {
            tail.next = newNode;
            newNode.prev = tail;
            tail = newNode;
        }
    }

    public void insertAfter(String keyNim, Mahasiswa06 data) {
        Node06 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;
        }

        Node06 newNode = new Node06(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() {
        Node06 current = head;
        while (current != null) {
            current.data.tampilInformasi();
            current = current.next;
        }
    }

    public void removeFirst() {
        if (isEmpty()) {
            System.out.println("Linked List masih kosong, tidak dapat
dihapus!");
        } else 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 Node06 search(String targetNim) {
        Node06 current = head;
        while (current != null) {
            if (current.data.nim.equals(targetNim)) {
                return current;
            }
            current = current.next;
        }
        return null;
    }
}
```

DllMain

```
\import java.util.Scanner;

public class DllMain {
    static Scanner sc = new Scanner(System.in);

    public static Mahasiswa06 inputMahasiswa() {
        System.out.print("Nim : ");
        String nim = sc.nextLine();
        System.out.print("Nama : ");
        String nama = sc.nextLine();
        System.out.print("Kelas : ");
        String kelas = sc.nextLine();
        System.out.print("IPK : ");
        double ipk = sc.nextDouble();
        sc.nextLine();
        Mahasiswa06 mhs = new Mahasiswa06(nim, nama, kelas, ipk);
        return mhs;
    }

    public static void main(String[] args) {
        DoubleLinkedList06 list = new DoubleLinkedList06();
        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("3. Hapus di awal");
            System.out.println("4. Hapus di akhir");
            System.out.println("5. Tampilkan data");
            System.out.println("6. Cari Mahasiswa berdasarkan NIM");
            System.out.println("0. Keluar");
            System.out.print("Pilih menu: ");
            pilihan = sc.nextInt();
            sc.nextLine();

            switch (pilihan) {
                case 1: {
                    Mahasiswa06 mhs = inputMahasiswa();
                    list.addFirst(mhs);
                    break;
                }
                case 2: {
                    Mahasiswa06 mhs = inputMahasiswa();
                    list.addLast(mhs);
                    break;
                }
                case 3:
                    list.removeFirst();
                    break;
                case 4:
                    list.removeLast();
                    break;
                case 5:
                    list.print();
                    break;
```

```
     break;
                case 6: {
                    System.out.print("Masukkan NIM yang dicari: ");
                    String nim = sc.nextLine();
                    Node06 found = list.search(nim);
                    if (found != null) {
                        System.out.println("Data ditemukan");
                        found.data.tampil();
                    } else {
                        System.out.println("Data tidak ditemukan");
                    }
                    break;
                }
                case 0:
                    System.out.println("Keluar dari program.");
                    break;
                default:
                    System.out.println("Pilihan tidak valid!");
            }
        } while (pilihan != 0);
    }
}
```

2. Hasil dari kode program

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
0. Keluar
Pilih menu: 1
NIM : 123
Nama : burhan
Kelas : 1e
IPK : 3.5

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
0. Keluar
Pilih menu: 5
NIM: 123, Nama: burhan, Kelas: 1e, IPK 3.5
```

PERTANYAAN

1. Single Linked List dan Double Linked List dibedakan dengan jumlah pointer yang ada di objek Node, seperti namanya Single Linked List hanya memiliki 1 attribute pointer yaitu next yang menandai objek setelah node itu sendiri sedangkan Double Linked List memiliki prev yang menandai objek sebelum node.
2. next berfungsi untuk menandai objek setelah node itu sendiri sedangkan prev berfungsi untuk menandai objek sebelum node. Keduanya merupakan pointer
3. konstruktor tersebut digunakan untuk menginisasi attribute head dan tail menjadi null, yang menandakan bahwa linked list tersebut masih kosong.
4. jika kondisi isEmpty() terpenuhi, attribute head dan tail akan langsung diisi dengan newNode, karena jika datanya hanya 1, otomatis value dari head dan tail akan sama
5. head.prev merupakan attribut pada node head (data pertama) yang menunjuk node sebelum data node tersebut, dengan statement tersebut, nilai dari head.prev akan diisi dengan newNode yang berarti newNode akan ditaruh di data paling depan.
6. Modifikasi fungsi `print()`

```
void print() {
      Node06 temp = head;
      if (isEmpty()) {
         System.out.println("Data kosong!");
         return;
      }
      while (temp != null) {
         temp.data.tampilInformasi();
         temp = temp.next;
      }
   }
```

```
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
Data Kosong!
```

7. current.next.prev = newNode, kode program ini berguna untuk mengisi attribut prev pada node setelahcurrentsehingga menghubungkan data setelahcurrentdengannewNode

8. Modifikasi Program

- DoubleLinkedList

```
void insertAfter(String key, Mahasiswa06 data) {
        Node06 newNode = new Node06(data);
        Node06 temp = head;
        while (temp != null && !temp.data.nim.equals(key)) {
            temp = temp.next;
        }

        if (temp == null) {
            System.out.println("Node dengan NIM " + key + " tidak
ditemukan");
        }

        if (temp == tail) {
            addLast(data);
        } else {
            newNode.next = temp.next;
            newNode.prev = temp;
            temp.next.prev = newNode;
            temp.next = newNode;
        }
    }
```

- DllMain

```
  case 7:
                {
                System.out.print("Masukkan nama mahasiswa : ");
                String namaMhs = scan.next();
```

PERCOBAAN 2

1. Kode program

```
Mahasiswa25 removeFirst() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus");
            return null;
        }
        if (head == tail) {
            Mahasiswa25 data = head.data;
            head = tail = null;
            return data;
        } else {
            Mahasiswa25 data = head.data;
            head = head.next;
            head.prev = null;
            return data;

        }
    }

    Mahasiswa25 removeLast() {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus");
            return null;
        }
        if (head == tail) {
            Mahasiswa25 data = tail.data;
            head = tail = null;
            return data;
        } else {
            Mahasiswa25 data = tail.data;
            tail = tail.prev;
            tail.next = null;
            return data;
        }
    }
```

2. Hasil dari kode program

```
Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
7. Tambah Data (Setelah Mahasiswa)
0. Keluar
Pilih menu: 2
NIM : 123
Nama : burhan
Kelas : 1e
IPK : 3.8

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
7. Tambah Data (Setelah Mahasiswa)
0. Keluar
Pilih menu: 3
Data sudah berhasil dihapus. Data yang terhapus adalah burhan

Menu Double Linked List Mahasiswa
1. Tambah di awal
2. Tambah di akhir
3. Hapus di awal
4. Hapus di akhir
5. Tampilkan data
7. Tambah Data (Setelah Mahasiswa)
0. Keluar
Pilih menu: 5
Data Kosong!
```

PERTANYAAN

1. mengubah value dari attribute head menjadi node setelahnya yaitu dengan mengakses head.next dan mengubah value head.prev yang merupakan pointer ke node sebelumnya menjadi null
2. modifikasi kode program
    - removeFirst

```
Mahasiswa06 removeFirst() {
       if (isEmpty()) {
           System.out.println("List kosong, tidak bisa dihapus");
           return null;
       }
       if (head == tail) {
           Mahasiswa06 data = head.data;
           head = tail = null;
           return data;
       } else {
           Mahasiswa06 data = head.data;
           head = head.next;
           head.prev = null;
           return data;

       }
   }
```

    - removeLast

```
Mahasiswa06 removeLast() {
       if (isEmpty()) {
           System.out.println("List kosong, tidak bisa dihapus");
           return null;
       }
       if (head == tail) {
           Mahasiswa06 data = tail.data;
           head = tail = null;
           return data;
       } else {
           Mahasiswa06 data = tail.data;
           tail = tail.prev;
           tail.next = null;
           return data;
       }
   }
```

TUGAS

1. fungsi add

```
void add(Mahasiswa06 data, int index) {
        if (index == 0) {
            addFirst(data);
            return;
        }

        Node06 temp = head;
        for (int i = 0; i < index - 1; i++) {
        if (temp == null) {
        System.out.println("Index melebihi panjang list");
        return;
        }
        temp = temp.next;
        }

        if (temp.next == null) {
            addLast(data);
            return;
        }

        Node06 newNode = new Node06(data);
        temp.next.prev = newNode;
        newNode.next = temp.next;
        temp.next = newNode;
        newNode.prev = temp;
    }
```

hasil dari kode program

```
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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
Pilih menu: 8
NIM : 321
Nama : ihsan
Kelas : 3e
IPK : 4.0
Masukkan Index : 2

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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
Pilih menu: 5
NIM: 123, Nama: burhan, Kelas: 1e, IPK 3.5
NIM: 231, Nama: udin, Kelas: 1e, IPK 3.7
NIM: 321, Nama: ihsan, Kelas: 3e, IPK 4.0
```

2. Fungsi RemoveAfter

```
 void removeAfter(String key) {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus");
            return;
        }
        Node06 temp = head;

        while (temp != null && !temp.data.nama.equalsIgnoreCase(key)) {
        temp = temp.next;
        }

        if (temp == null || temp.next == null) {
        System.out.println("Node setelah \"" + key + "\" tidak ditemukan atau
tidak ada");
        return;
        }

        temp.next.prev = temp;
        temp.next = temp.next.next;

    }
```

Hasil kode program

```
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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
Pilih menu: 5
NIM: 123, Nama: burhan, Kelas: 1e, IPK 3.5
NIM: 231, Nama: udin, Kelas: 1e, IPK 3.7
NIM: 321, Nama: ihsan, Kelas: 3e, IPK 4.0

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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
Pilih menu: 9
Masukkan nama mahasiswa : hani
Node setelah "hani" tidak ditemukan atau tidak ada

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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
Pilih menu: 9
Masukkan nama mahasiswa : burhan

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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
Pilih menu: 5
NIM: 123, Nama: burhan, Kelas: 1e, IPK 3.5
NIM: 321, Nama: ihsan, Kelas: 3e, IPK 4.0
```

3. Fungsi Remove

```
void remove(int index) {
        if (isEmpty()) {
            System.out.println("List kosong, tidak bisa dihapus");
            return;
        }

        if (index < 0) {
        System.out.println("Index tidak valid");
        return;
    }

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

        Node06 temp = head;

        for(int i = 0; i < index; i++) {
        if (temp == null) {
            System.out.println("Index melebihi panjang list");
            return;
        }
            temp = temp.next;
        }

        if (temp.next == null) {
            removeLast();
            return;
        }

        temp.next.prev = temp.prev;
        temp.prev.next = temp.next;
    }
```

Hasil dari kode program

```
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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
9. Hapus Data Mahasiswa (Setelah Mahasiswa)
10. Hapus Data (Index Tertentu)
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data (Index Tertentu)
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 10
Masukkan index : 2
Data sudah berhasil dihapus. Data yang terhapus adalah ihsan
```

4. getFirst, getLast, getIndex

```
Mahasiswa06 getFirst() {
        if (isEmpty()) {
            return null;
        }
        return head.data;
    }

    Mahasiswa06 getLast() {
        if (isEmpty()) {
            return null;
        }

        return tail.data;
    }

    Mahasiswa06 getIndex(int index) {
        if (isEmpty()) {
            return null;
        }

        Node06 temp = head;

        for(int i = 0; i < index; i++) {
        if (temp == null) {
            System.out.println("Index melebihi panjang list");
            return null;
        }
            temp = temp.next;
        }
        return temp.data;
    }
```

hasil dari kode program

```
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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
9. Hapus Data Mahasiswa (Setelah Mahasiswa)
10. Hapus Data (Index Tertentu)
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data (Index Tertentu)
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 11
NIM: 231, Nama: udin, Kelas: 2e, IPK 3.7

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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
9. Hapus Data Mahasiswa (Setelah Mahasiswa)
10. Hapus Data (Index Tertentu)
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data (Index Tertentu)
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 12
NIM: 123, Nama: burha, Kelas: 1e, IPK 3.6
```

```
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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
9. Hapus Data Mahasiswa (Setelah Mahasiswa)
10. Hapus Data (Index Tertentu)
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data (Index Tertentu)
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 13
Masukkan Index : 2
NIM: 121, Nama: ihsan, Kelas: 1e, IPK 3.6
```

5. getSize

```
int getSize() {
    int counter = 0;

    Node06 temp = head;
    while (temp != null) {
        temp = temp.next;
        counter++;
    }

    return counter;
}
```

Hasil dari kode program

```
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
7. Tambah Data (Setelah Mahasiswa)
8. Tambah data (Index Tertentu)
9. Hapus Data Mahasiswa (Setelah Mahasiswa)
10. Hapus Data (Index Tertentu)
11. Tampilkan data pertama
12. Tampilkan data terakhir
13. Tampilkan data (Index Tertentu)
14. Tampilkan ukuran list
0. Keluar
Pilih menu: 14
Jumlah Mahasiswa : 3
```