PRAKTIKUM ALGORITMA DAN STRUKTUR DATA

JOBSHEET PERTEMUAN KE-10

NAMA : ALVINO VALERIAN D.R

KELAS : 1A

NO. ABSEN : 05

NIM : 2341720027

PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2024

12.2.2 Verifikasi Hasil Percobaan

```
lingked list kosong!
Size :0
=========================================================
7       3       4
berhasil diisi!
Size :3
=========================================================
7       40      3       4
berhasil diisi!
Size :4
=========================================================
lingked list kosong!
Size :0
=========================================================

D:\alvino\Semester 2\Prak algorita & struktur data\jobsheet10>
```

12.2.3 Pertanyaan Percobaan

1. Jelaskan perbedaan antara single linked list dengan double linked lists!

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

3. Perhatikan konstruktor pada class DoubleLinkedLists. Apa kegunaan inisialisasi atribut head dan size seperti pada gambar berikut ini?

```
public DoubleLinkedLists() {
    head = null;
    size = 0;
}
```

4. Pada method addFirst(), kenapa dalam pembuatan object dari konstruktor class Node prev dianggap sama dengan null?

Node newNode = new Node(null, item, head);

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

6. Perhatikan isi method addLast(), apa arti dari pembuatan object Node dengan mengisikan parameter prev dengan current, dan next dengan null?

Node newNode = new Node(current, item, null);

7. Pada method add(), terdapat potongan kode program sebagai berikut:

```
while (i < index) {
    current = current.next;
    i++;
}
if (current.prev == null) {
    Node newNode = new Node(null, item, current);
    current.prev = newNode;
    head = newNode;
} else {
    Node newNode = new Node(current.prev, item, current);
    newNode.prev = current.prev;
    newNode.next = current;
    current.prev.next = newNode;
    current.prev = newNode;
}
```

jelaskan maksud dari bagian yang ditandai dengan kotak kuning.

JAWABAN:

1. Single linked list lebih sederhana dan lebih hemat memori, tetapi operasi tertentu seperti pencarian dan penghapusan di tengah list menjadi lebih rumit dan memakan waktu.

Double linked list menyediakan akses maju dan mundur yang lebih mudah, membuatnya lebih efisien untuk operasi tertentu, meskipun memerlukan lebih banyak ruang dalam memori.

Perbedaanya hanya pada node jika double bisa next dan prev

2. Atribut next: Atribut ini menunjukkan node berikutnya dalam linked list. Dengan adanya atribut next, Anda dapat bergerak maju dari satu node ke node berikutnya dalam linked list.

Atribut prev: Atribut ini menunjukkan node sebelumnya dalam linked list. Dengan adanya atribut prev, Anda dapat bergerak mundur dari satu node ke node sebelumnya dalam linked list.

3. Inisialisasi atribut head: Dengan mengatur head menjadi null, konstruktor menandakan bahwa saat objek double linked list pertama kali dibuat, tidak ada node yang ada dalam linked list tersebut.

   Inisialisasi atribut size: Dengan mengatur size menjadi 0, konstruktor menandakan bahwa linked list tidak memiliki elemen pada awalnya.

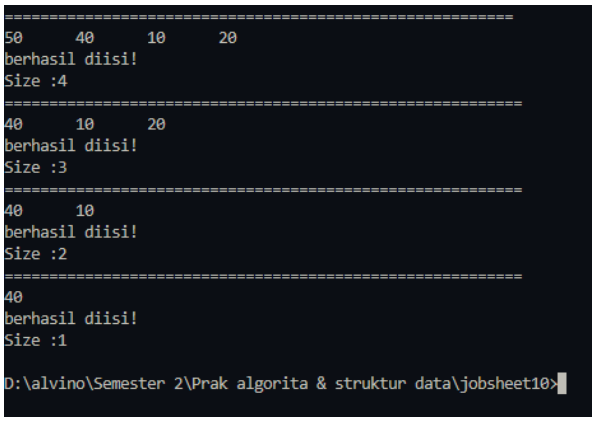
4. ketika membuat objek node baru menggunakan konstruktor kelas Node05, atribut prev diatur ke null karena node baru tersebut akan menjadi node pertama dalam linked list. Dengan kata lain, tidak ada node sebelumnya yang perlu ditunjukkan oleh atribut prev. Oleh karena itu, dengan mengatur prev ke null, kita menandakan bahwa node baru adalah node pertama dalam linked list tersebut.

5. head.prev = newNode bertujuan untuk memperbarui pointer prev dari node yang sebelumnya menjadi node baru yang ditambahkan.

6. pembuatan objek Node dengan mengisikan parameter prev dengan current dan next dengan null bertujuan untuk membuat node baru yang akan ditambahkan di akhir linked list. Elemen baru akan menjadi node terakhir

7. Kondisi if (current.prev == null) memeriksa apakah kita berada di awal linked list. Pembuatan node baru newNode dengan prev diatur ke null dan next ke current. Mengatur prev dari node pertama (current) menjadi node baru. Memperbarui head agar menunjuk ke node baru yang ditambahkan.

12.3.2 Verifikasi Hasil Percobaan



12.3.3 Pertanyaan Percobaan

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

head = head.next;

head.prev = null;

2. Bagaimana cara mendeteksi posisi data ada pada bagian akhir pada method removeLast()?

3. Jelaskan alasan potongan kode program di bawah ini tidak cocok untuk perintah remove!

```
Node tmp = head.next;

head.next=tmp.next;
tmp.next.prev=head;
```

4. Jelaskan fungsi kode program berikut ini pada fungsi remove!

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

JAWABAN

1.Statement head = head.next; digunakan untuk memperbarui head sehingga menunjuk ke node kedua dalam linked list setelah node pertama dihapus. Ini menghapus node pertama dari linked list.

Statement head.prev = null; digunakan untuk memastikan bahwa prev dari node pertama yang baru (yang sekarang menjadi node kedua setelah penghapusan) diatur ke null, karena tidak ada node sebelumnya setelah penghapusan node pertama.

2. memeriksa apakah head.next == null, dapat mendeteksi posisi data pada bagian akhir dalam method removeLast().

3. Mengasumsikan bahwa elemen yang ingin dihapus selalu berada di posisi kedua dalam linked list, yang tidak benar.Tidak memperhitungkan situasi ketika head.next adalah null, yang dapat menyebabkan NullPointerException.Tidak melakukan penghapusan sebenarnya dari linked list, hanya mengubah pointer, yang dapat menyebabkan kebocoran memori dan kesalahan dalam operasi berikutnya pada linked list.

4. current.next.prev = current.prev;: Pernyataan ini bertujuan untuk mengubah pointer prev dari node setelahnya (current.next) sehingga menunjuk ke node sebelumnya (current.prev). Dengan kata lain, kita memperbarui pointer prev dari node setelah node yang ingin dihapus untuk menunjuk ke node sebelumnya, sehingga menjaga konsistensi struktur double linked list.

12.4.2 Verifikasi Hasil Percobaan

```
lingked list kosong!
Size :0
=====================================================
7       3       4
berhasil diisi!
Size :3
=====================================================
7       40      3       4
berhasil diisi!
Size :4
=====================================================
Data awal pada lingked lists adalah: 7
Data akhir pada lingked lists adalah: 4
Data indeks ke-1 pada lingked lists adalah: 40
```

12.4.3 Pertanyaan Percobaan

1. Jelaskan method **size()** pada class DoubleLinkedLists!

2. Jelaskan cara mengatur indeks pada double linked lists supaya dapat dimulai dari indeks ke-1!

3. Jelaskan perbedaan karakteristik fungsi **Add** pada Double Linked Lists dan Single Linked Lists!

4. Jelaskan perbedaan logika dari kedua kode program di bawah ini!

```
public boolean isEmpty(){
    if(size ==0){
        return true;
    } else{
        return false;
    }
}
```

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

JAWABAN

1. Method size() pada DoubleLinkedLists:

Mengembalikan jumlah elemen dalam linked list.Implementasinya sederhana, hanya mengembalikan nilai dari atribut size.Berguna untuk memantau dan mengelola ukuran linked list.

2. Atur Indeks pada Double Linked Lists untuk Dimulai dari 1:

Sesuaikan logika operasi indeks pada metode seperti add(), remove(), dll.Geser nilai indeks yang diterima dalam metode sesuai kebutuhan, misalnya, kurangi 1 dari nilai indeks.

3. Perbedaan Fungsi Add pada Double Linked Lists dan Single Linked Lists:

Double Linked Lists:

Memerlukan pengaturan dua pointer: untuk node sebelumnya dan node berikutnya.Memungkinkan penggunaan metode add() tanpa traversal lengkap untuk menambahkan elemen di mana saja dalam linked list.

Single Linked Lists:

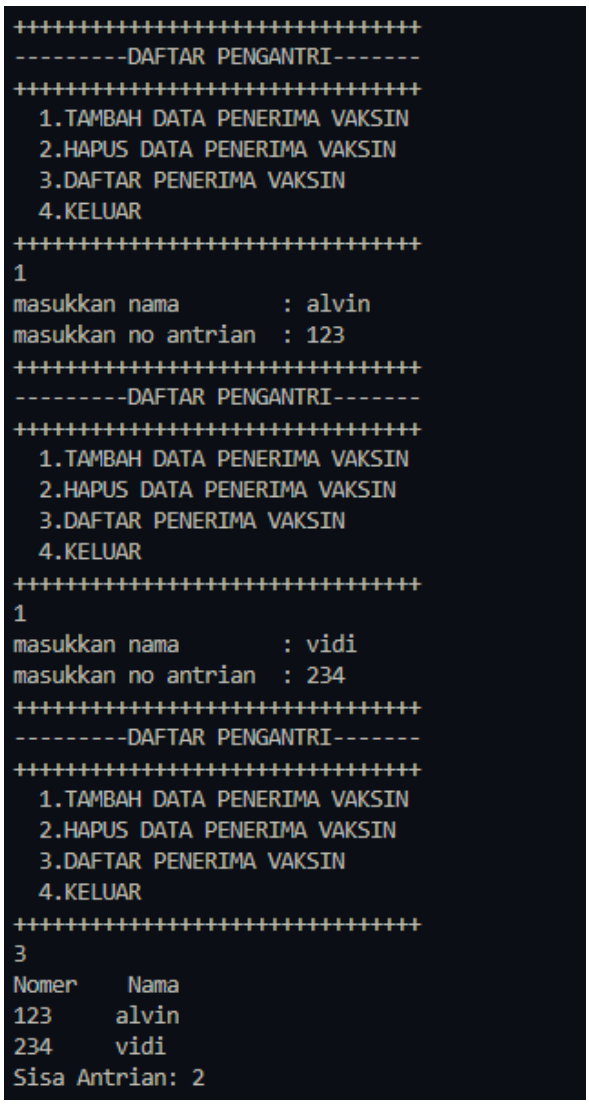
Hanya memerlukan pengaturan satu pointer: untuk node berikutnya.Diperlambat dalam operasi tambah di tengah linked list karena memerlukan traversal untuk mencari node sebelumnya.

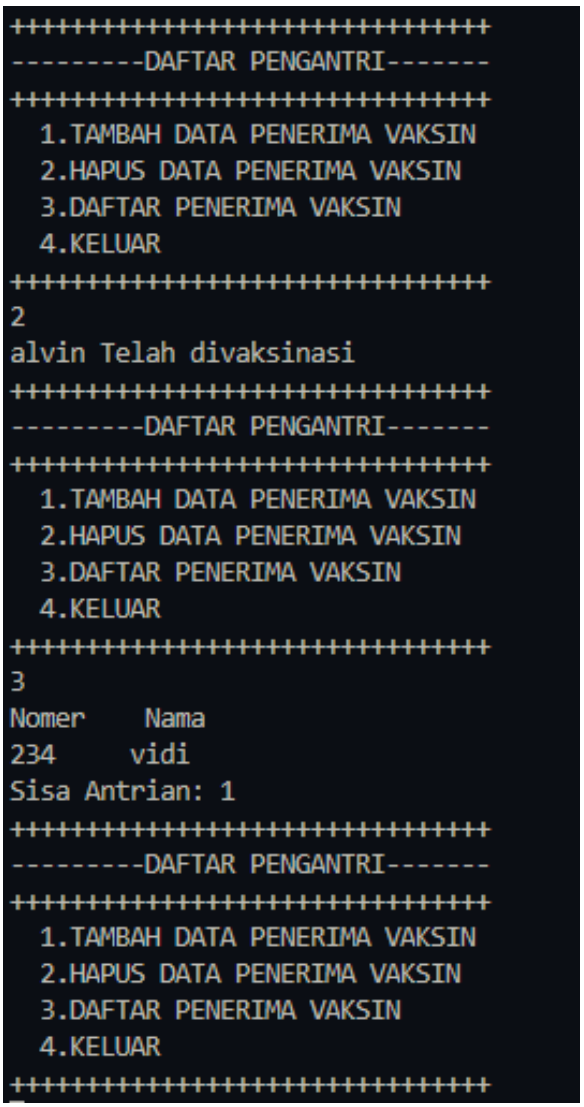
4. Metode pertama (isEmpty() dengan perbandingan ukuran) memeriksa kekosongan linked list berdasarkan jumlah elemen yang disimpan dalam atribut size.

Metode kedua (isEmpty() dengan perbandingan head) memeriksa kekosongan linked list berdasarkan keberadaan node pertama (head).

Kedua metode ini sama-sama efektif dalam menentukan apakah linked list kosong atau tidak, tetapi metode kedua (isEmpty() dengan perbandingan head) cenderung lebih umum digunakan dan lebih efisien karena tidak melibatkan perhitungan ukuran size.

TUGAS1





```
package tugas1;

public class Vaksin05 {
    int noAntri;
    String nama;
    Vaksin05 prev,next;

    Vaksin05(int noAntri,String nama,Vaksin05 prev,Vaksin05 next){
        this.nama=nama;
        this.noAntri=noAntri;
        this.prev=prev;
        this.next=next;
    }
}
```

```
package tugas1;

public class Dll05 {
    Vaksin05 head;
    int size;

    public Dll05(){
        head=null;
        size=0;
    }
    public boolean isEmpty(){
        return head==null;
    }
    public void addFirst(int noAntri,String nama){
        if (isEmpty()) {
            head = new Vaksin05(noAntri, nama, null, null);
        }else{
            Vaksin05 node = new Vaksin05(noAntri, nama, null, head);
            head.prev = node;
            head = node;
        }
        size++;
```

```
    }
    public void addLast(int noAntri,String nama){
        if (isEmpty()) {
            addFirst(noAntri, nama);;
        }else{
            Vaksin05 current = head;
            while (current.next != null) {
                current = current.next;
            }
            Vaksin05 newNode = new Vaksin05(noAntri, nama, current,
null);
            current.next = newNode;
            size++;
        }
    }
public int size(){
    return size;
}
public void clear(){
    head = null;
    size=0;
}
public void print(){
    if (!isEmpty()) {
        Vaksin05 tmp = head;
        System.out.println("Nomer\t Nama\t");
        while (tmp != null) {
            System.out.println(tmp.noAntri+"\t" + tmp.nama+"  \t");
            tmp=tmp.next;
        }
        System.out.println("Sisa Antrian: "+size);
    }else{
        System.out.println("Tidak ada Antrian");
    }
}
public void removeFirst() throws Exception{
    if (isEmpty()) {
```

```
        throw new Exception("tidak ada yang mengantri");

    }else if (size == 1) {
        removeLast();
    }else{
        head=head.next;
        Vaksin05 penerima = head.prev;
        System.out.println(penerima.nama+ " Telah divaksinasi");
        head.prev=null;
        size--;
    }
}
public void removeLast() throws Exception{
    if (isEmpty()) {
        throw new Exception("tidak ada yang mengantri");

    }else if (head.next == null) {
        Vaksin05 penerima = head;
        System.out.println(penerima.nama+" Telah divaksinasi");
        head=null;
        size--;
        return;
    }
}
}
```

```
package tugas1;
import java.util.Scanner;;

public class DllMain05 {

    public static void main(String[] args)throws Exception {
        Scanner sc = new Scanner(System.in);
        Dll05 pasien05 = new Dll05();

        int menu;
        do {
```

```
            System.out.println("++++++++++++++++++++++++++++++++");
            System.out.println("---------DAFTAR PENGANTRI-------");
            System.out.println("++++++++++++++++++++++++++++++++");
            System.out.println("  1.TAMBAH DATA PENERIMA VAKSIN ");
            System.out.println("  2.HAPUS DATA PENERIMA VAKSIN  ");
            System.out.println("  3.DAFTAR PENERIMA VAKSIN      ");
            System.out.println("  4.KELUAR                      ");
            System.out.println("++++++++++++++++++++++++++++++++");
            menu =sc.nextInt();
            sc.nextLine();
            switch (menu) {
                case 1:
                    System.out.print("masukkan nama        : ");
                    String nama = sc.nextLine();
                    System.out.print("masukkan no antrian  : ");
                    int noAntri =sc.nextInt();
                    pasien05.addLast(noAntri, nama);
                    break;
                case 2:
                    pasien05.removeFirst();
                    break;
                case 3:
                    pasien05.print();
                    break;
                case 4:
                    System.exit(0);
                    break;

                default:
                System.out.println("input salah!");
                    break;
            }
        } while (menu !=4);
        sc.close();
    }

}
```

TUGAS2

```
package tugas2;

public class film05 {
    String nama;
    int ID;
    float rating;
    film05 prev,next;

    film05(String nama,int ID,float rating,film05 prev,film05 next){
        this.nama=nama;
        this.ID=ID;
        this.rating=rating;
        this.prev=prev;
        this.next=next;
    }
}
```

```
package tugas2;
public class DllF05 {
    film05 head;
    int size;

    public DllF05(){
        head = null;
        size = 0;
    }
    public boolean isEmpty(){
        return head == null;
    }
    public void addFirst(String nama,int ID,float rating){
        if (isEmpty()) {
            head = new film05(nama, ID, rating, null, null);
        }else{
            film05 newNode = new film05(nama, ID, rating, null,
head);
            head.prev = newNode;
```

```
            head = newNode;
        }
        size++;
    }
    public void addLast(String nama,int ID,float rating){
        if (isEmpty()) {
            addFirst(nama, ID, rating);
            film05 current = head;
            while (current.next != null) {
                current = current.next;
            }
            film05 newNode = new film05(nama, ID, rating, current,
null);
            current.next = newNode;
            size++;
        }
    }
    public void add(float rating,int ID,String nama,int index )
throws Exception{
        if (isEmpty()) {
            addFirst(nama, ID, rating);
        }else if (index < 0 || index > size){
            throw new Exception("Nilai indeks di luar batas");
        }else{
            film05 current = head;
            int i = 0;
            while (i < index) {
                current = current.next;
                i++;
            }
            if (current.prev == null) {
              film05 newNode = new film05(nama, ID, rating, null,
current);
              current.prev = newNode;
              head=newNode;
            }else{
```

```
                film05 newNode = new film05(nama, ID, rating,
current.prev, current);
                newNode.prev = current.prev;
                newNode.next = current;
                current.prev.next= newNode;
                current.prev = newNode;
            }

        }
        size++;
    }
    public int size(){
        return size;
    }
    public void clear(){
        head = null;
        size=0;
    }
    public void print(){
        if (!isEmpty()) {
            film05 tmp = head;
            while (tmp != null) {
                System.out.println("Judul: " + tmp.nama + "\tRating:
" + tmp.rating + "\tID: " + tmp.ID);
                tmp=tmp.next;

            }
            System.out.println("\nberhasil diisi!");
        }else{
            System.out.println("lingked list kosong!");
        }
    }
    public void removeFirst() throws Exception{
        if (isEmpty()) {
            throw new Exception("Lingked list masih kosong");

        }else if (size == 1) {
```

```
            removeLast();
        }else{
            head=head.next;
            head.prev=null;
            size--;
        }
    }
    public void removeLast() throws Exception{
        if (isEmpty()) {
            throw new Exception("Lingked list masih kosong");

        }else if (head.next == null) {
            head = null;
            size--;
            return;
        }
        film05 current = head;
        while (current.next.next !=null) {
            current = current.next;
        }
        current.next = null;
        size--;
    }
    public void remove(int index) throws Exception{
        if (isEmpty() || index >=size ) {
            throw new Exception("Nilai indeks di luar batas");

        }else if (index==0) {
            removeFirst();
            }else{
                film05 current = head;
                int i = 0;
                while (i < index) {
                    current = current.next;
                    i++;
            }
            if (current.next == null) {
```

```
                current.prev.next = null;
            }else if (current.prev == null) {
                current = current.next;
                current.prev=null;
                head = current;
            }else{
                current.prev.next = current.next;
                current.next.prev = current.prev;
            }
            size--;
        }
    }
    public void sortRatingDescending() {
        film05 current = head;
        film05 next = null;
        while (current != null) {
            next = current.next;
            while (next != null) {
                if (current.rating < next.rating) {
                    float tmp = current.rating;
                    current.rating = next.rating;
                    next.rating = tmp;
                }
                next = next.next;
            }
            current = current.next;
        }
    }
    public film05 search(int id){
        if (isEmpty()) {
            System.out.println("Data masih kosong");
            return null;
        }
        film05 current = head;
        while (current != null) {
            if (current.ID == id) {
                return current;
```

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

    public void printSearch(int id){
        film05 Search = search(id);
        if (
            Search != null) {
            System.out.println("Film Ditemukan");
            System.out.println("Id Film     : " + Search.ID);
            System.out.println("Judul Film  : " + Search.nama);
            System.out.println("Rating Film : " + Search.rating);
        } else {
            System.out.println("Id Film Tidak Ditemukan");
        }
    }
}
```

```
package tugas2;
import java.util.Scanner;

public class filmMain05 {

    public static void main(String[] args) throws Exception{
        Scanner sc = new Scanner(System.in);
        DllF05 film05 = new DllF05();
        int menu;
        do {
            System.out.println("================================");
            System.out.println("DATA FILM LAYAR LEBAR");
            System.out.println("================================");
            System.out.println("  1.TAMBAH DATA AWAL");
            System.out.println("  2.TAMBAH DATA AKHIR");
            System.out.println("  3.TAMBAH DATA INDEX TERTENTU");
            System.out.println("  4.HAPUS DATA PERTAMA");
```

```
            System.out.println("  5.HAPUS DATA TERAKHIR");
            System.out.println("  6.HAPUS DATA TERTENTU");
            System.out.println("  7.CETAK");
            System.out.println("  8.CARI ID FILM");
            System.out.println("  9.URUT DATA RATING FILM-DESC");
            System.out.println("  10.KELUAR");
            System.out.println("================================");
            menu =sc.nextInt();
            sc.nextLine();
            switch (menu) {
                case 1:
                    System.out.println("Masukkan data film poisi
awal");
                    System.out.print("ID Film: ");
                    int ID =sc.nextInt();
                    System.out.print("Judul Film: ");
                    sc.nextLine();
                    String judul= sc.nextLine();
                    System.out.print("Rating film: ");
                    float rating = sc.nextFloat();
                    film05.addFirst(judul, ID, rating);
                    break;
                case 2:
                System.out.println("Masukkan data film poisi awal");
                System.out.print("ID Film: ");
                int ID1 =sc.nextInt();
                System.out.print("Judul Film: ");
                sc.nextLine();
                String judul1= sc.nextLine();
                System.out.print("Rating film: ");
                float rating1 = sc.nextFloat();
                film05.addLast(judul1, ID1, rating1);
                    break;
                case 3:
                System.out.println("Masukkan data film poisi awal");
                System.out.print("ID Film: ");
                int ID2 =sc.nextInt();
```

```
                System.out.print("Judul Film: ");
                sc.nextLine();
                String judul2= sc.nextLine();
                System.out.print("Rating film: ");
                float rating2 = sc.nextFloat();
                System.out.print("Indeks ke- ");
                int index =sc.nextInt();
                film05.add(rating2, ID2, judul2, index);
                    break;
                case 4:
                    film05.removeFirst();
                    break;
                case 5:
                    film05.removeLast();
                    break;
                case 6:
                System.out.print("Hapus film urutan ke- ");
                    int index3 = sc.nextInt();
                    film05.remove(index3);
                    break;
                case 7:
                    film05.print();
                    break;
                case 8:
                    System.out.println("film yang akan dicari
berdasarkan ID");
                    System.out.print("ID film: ");
                    int ID3 =sc.nextInt();
                    film05.search(ID3);
                    film05.printSearch(ID3);
                    break;
                case 9:
                    film05.sortRatingDescending();
                    break;
                case 10:
                    System.exit(0);
                    break;
```

```
                default:
                    break;
            }
        } while (menu !=10);
        sc.close();
    }
}
```

```
================================
DATA FILM LAYAR LEBAR
================================
  1.TAMBAH DATA AWAL
  2.TAMBAH DATA AKHIR
  3.TAMBAH DATA INDEX TERTENTU
  4.HAPUS DATA PERTAMA
  5.HAPUS DATA TERAKHIR
  6.HAPUS DATA TERTENTU
  7.CETAK
  8.CARI ID FILM
  9.URUT DATA RATING FILM-DESC
  10.KELUAR
================================
1
Masukkan data film poisi awal
ID Film: 1
Judul Film: spiderman
Rating film: 3,3
```

```
================================
DATA FILM LAYAR LEBAR
================================
  1.TAMBAH DATA AWAL
  2.TAMBAH DATA AKHIR
  3.TAMBAH DATA INDEX TERTENTU
  4.HAPUS DATA PERTAMA
  5.HAPUS DATA TERAKHIR
  6.HAPUS DATA TERTENTU
  7.CETAK
  8.CARI ID FILM
  9.URUT DATA RATING FILM-DESC
  10.KELUAR
================================
2
Masukkan data film poisi awal
ID Film: 3
Judul Film: upin ipin
Rating film: 4,5
================================
DATA FILM LAYAR LEBAR
================================
  1.TAMBAH DATA AWAL
  2.TAMBAH DATA AKHIR
  3.TAMBAH DATA INDEX TERTENTU
  4.HAPUS DATA PERTAMA
  5.HAPUS DATA TERAKHIR
  6.HAPUS DATA TERTENTU
  7.CETAK
  8.CARI ID FILM
  9.URUT DATA RATING FILM-DESC
  10.KELUAR
================================
1
Masukkan data film poisi awal
ID Film: 4
Judul Film: no way home
```

```
================================
DATA FILM LAYAR LEBAR
================================
  1.TAMBAH DATA AWAL
  2.TAMBAH DATA AKHIR
  3.TAMBAH DATA INDEX TERTENTU
  4.HAPUS DATA PERTAMA
  5.HAPUS DATA TERAKHIR
  6.HAPUS DATA TERTENTU
  7.CETAK
  8.CARI ID FILM
  9.URUT DATA RATING FILM-DESC
  10.KELUAR
================================
1
Masukkan data film poisi awal
ID Film: 2
Judul Film: batman
Rating film: 2,2
```

```
================================
DATA FILM LAYAR LEBAR
================================
  1.TAMBAH DATA AWAL
  2.TAMBAH DATA AKHIR
  3.TAMBAH DATA INDEX TERTENTU
  4.HAPUS DATA PERTAMA
  5.HAPUS DATA TERAKHIR
  6.HAPUS DATA TERTENTU
  7.CETAK
  8.CARI ID FILM
  9.URUT DATA RATING FILM-DESC
  10.KELUAR
================================
7
Judul: no way home      Rating: 6.7     ID: 4
Judul: batman   Rating: 2.2     ID: 2
Judul: spiderman        Rating: 3.3     ID: 1
```

```
================================
  1.TAMBAH DATA AWAL
  2.TAMBAH DATA AKHIR
  3.TAMBAH DATA INDEX TERTENTU
  4.HAPUS DATA PERTAMA
  5.HAPUS DATA TERAKHIR
  6.HAPUS DATA TERTENTU
  7.CETAK
  8.CARI ID FILM
  9.URUT DATA RATING FILM-DESC
  10.KELUAR
================================
9
================================
DATA FILM LAYAR LEBAR
================================
  1.TAMBAH DATA AWAL
  2.TAMBAH DATA AKHIR
  3.TAMBAH DATA INDEX TERTENTU
  4.HAPUS DATA PERTAMA
  5.HAPUS DATA TERAKHIR
  6.HAPUS DATA TERTENTU
  7.CETAK
  8.CARI ID FILM
  9.URUT DATA RATING FILM-DESC
  10.KELUAR
================================
7
Judul: no way home      Rating: 6.7     ID: 4
Judul: batman   Rating: 3.3     ID: 2
Judul: spiderman        Rating: 2.2     ID: 1
```