

MODUL I

STACK DAN QUEUE

1.1 PERMASALAHAN

1.3.1 Hard Level

Dihari berikutnya Yanto kembali menjadi volunteer penjaga perpustakaan sedirian dikarenakan yanti tidak dapat hadir, tidak lama setelah Yanto memulai shift-nya beberapa orang mulai muncul untuk meminjam buku. Sesudah orang-orang itu memilih buku yang ingin mereka pinjam, mereka pun berbaris di depan meja Yanto dengan barisan seperti gambar disamping



Gambar 1.1 Permasalahan 1

Kazuma maju dan menunjukkan buku-buku yang ingin dia pinjam ke pada Yanto, Kazuma meminjam 2 buku yang berjudul "Belajar Java" dan "Cara Menjadi Orang Kaya", atribut lengkap bukunya dapat dilihat pada gambar disamping. Setelah Kazuma menyelesaikan peminjamannya, ia pun pergi. Sekarang tinggal tersisa 3 orang dalam antrian, dan antrian setelah Kazuma menjadi antrian pertama.

```

=====
=          BUKU KAZUMA          =
=====
Judul Buku : Cara Menjadi Orang Kaya
Pengarang : Teguh
Genre : Fantasi
Status Buku : Buku Biasa
=====

Judul Buku : Belajar Java
Pengarang : Raysen
Genre : Edukasi
Status Buku : Buku Biasa
=====

=====
=          DAFTAR ANTRIAN          =
=====
Nama : Hu Tao
Antrian ke : 1
Jumlah Buku : 3
Kartu Spesial : Ada
=====

Nama : Kafka
Antrian ke : 2
Jumlah Buku : 3
Kartu Spesial : Tidak ada
=====

Nama : Xiangling
Antrian ke : 3
Jumlah Buku : 1
Kartu Spesial : Tidak ada
=====

```

Gambar 1.1 Permasalahan 2

Sekarang giliran Hu Tao untuk menunjukkan buku yang ingin dia pinjam kepada Yanto, Hu Tao meminjam 3 buku yang berjudul "Cara Tidur Cepat", "Belajar C++" dan "Belajar Ilmu Hitam", Atribut lengkap bukunya dapat dilihat pada gambar disamping.

Di perpustakaan ini terdapat dua jenis buku, buku pertama adalah buku biasa dan buku kedua adalah buku terkutuk. Buku terkutuk adalah buku yang memiliki informasi tentang pengetahuanpengetahuan terlarang jadi untuk meminjam buku terkutuk, peminjam harus memiliki kartu spesial. Karena Hu Tao memiliki kartu spesial, dia dapat meminjam buku terkutuk dan setelah menyelesaikan peminjamannya, Hu Tao langsung pergi. Di antrian sekarang tertinggal 2 orang.

Setelah Hu Tao keluar, seseorang datang dengan terburu buru masuk ke dalam antrian. Orang itu adalah Sucrose, ia tampaknya ingin meminjam buku juga. di tangannya terdapat 3 tumpukkan buku.

Xiangling tiba tiba mendapat panggilan dari Zhongli, Yanto bisa mendengar samar samar percakapan xiangling di telefonnya, Yanto mendengar bahwa Gouba sedang mengamuk di Liyue dan butuh bantuan xiangling secepat mungkin. Karena urusan mendesak itu xiangling menyimpan Kembali bukunya dan langsung pergi. Tinggal tersisa dua orang di antrian. Sucrose juga nampaknya sedang terburu buru, ia pun menanyakan kepada kafka apakah dia boleh bertukar tempat dengan kafka. Kafka setuju, Sucrose dan Kafka pun bertukar tempat.

```
=====
=          DAFTAR ANTRIAN          =
=====
Nama : Sucrose
Antrian ke : 1
Jumlah Buku : 3
Kartu Spesial : Ada
=====
Nama : Kafka
Antrian ke : 2
Jumlah Buku : 3
Kartu Spesial : Tidak ada
=====
```

Gambar 1.1 Permasalahan 3

Sucrose menunjukkan buku-buku yang ingin ia pinjam ke Yanto, Sucrose meminjam 3 buku dengan judul "Resurrection", "Alhcemy" dan "Durin The Forgotten Dragon", Atribut lengkap bukunya dapat dilihat pada gambar disamping.

```

=====
                        BUKU SUCROSE
=====
Judul Buku : Durin The Forgotten Dragon
Pengarang : Gold
Genre : Misteri
Status Buku : Buku Biasa
=====

Judul Buku : Alhcemy
Pengarang : Albedo
Genre : Sience
Status Buku : Cursed
=====

Judul Buku : Resurrection
Pengarang : Unknown
Genre : Unknown
Status Buku : Cursed
=====

```

Gambar 1.1 Permasalahan 4

Setelah Sucrose menyelesaikan peminjamannya, ia pun pergi dengan terburu-buru. Sekarang tinggal tersisa 1 orang dalam antrian yaitu Kafka. Yanto memeriksa satu per-satu buku yang ingin di pinjam kafka, dan salah satu buku yang ingin kafka pinjam adalah buku terkutuk. Yanto menanyakan kepada Kafka apakah dia memiliki kartu special, dan kafka menjawab bahwa dia tidak memiliki kartu special. Yanto pun memberi tahu Kafka bahwa Kafka memerlukan kartu special untuk meminjam buku terkutuk. Yanto pun memindahkan buku yang terkutuk ke bagian paling atas tumpukkan dan meminta kafka untuk mengembalikan buku tersebut. Setelah kafka mengembalikan buku yang terkutuk, ia menyelesaikan proses peminjaman. Sudah tidak ada orang dalam antrian.

1.2 HASIL PERCOBAAN

1.2.1 Hard Level

1. Algoritma

- a. Langkah pertama membuat kelas *book*,
 - i. deklarasi variabel instance: *title*, *author*, genre dan status.
 - ii. Buat konstruktor untuk mengalokasikan data yang akan mengatur nilai tersebut saat node baru ditambahkan.
 - iii. Buat *display book* untuk menampilkan informasi buku.
- b. Langkah kedua membuat kelas *borrower*
 - i. Deklarasi atribut seperti nama, *specialcard*, *borrowerBooks*.

- ii. Buat konstruktor untuk mengalokasikan dan mengatur nilai dari atribut tersebut jika nilai di tambahkan.
 - iii. Fungsi `hasSpecialCard()`:, mengindikasikan apakah peminjam memiliki kartu spesial. Mengembalikan nama peminjam dengan `getName`.
 - iv. Buat `displayBorrower` untuk menampilkan informasi peminjam seperti nama, nomor antrian, jumlah buku dan status kartu spesial.
 - v. Buat `displayBook`, untuk menampilkan semua buku yang di pinjam.
 - vi. Buat `removeNonSpecialBook` untuk menghapus book dari daftar jika peminjam buku dengan status `cursed` karena tidak punya kartu spesial.
- c. Langkah selanjutnya membuat kelas `LibraryQueue` untuk mengimplementasi antrian perpustakaan.
- i. Buat objek `queue<Borrower>` untuk menyimpan daftar `Borrower` atau peminjam yang sedang dalam antrian.
 - ii. Buat konstruktor yang menginisialisasi `queue` sebagai instance dari `LinkedList` dengan tipe data `queue`.
 - iii. Buat `addBorrower`, untuk menambahkan peminjam dalam antrian dan objek peminjam di tambahkan di belakang antrian.
 - iv. Buat `removeBorrower` untuk menghapus peminjam yang berada di posisi pertama antrian
 - v. Buat `showQueue` untuk menampilkan seluruh daftar peminjam dalam antrian.
 - vi. Buat `displayBorrowerBooks` untuk menampilkan semua buku yang dipinjam berdasarkan nama yang dicari, jika ditemukan maka panggil fungsi `displayBooks` dari `borrower` tersebut untuk menampilkan buku yang dipinjam.
 - vii. Buat `swapQueue` untuk menukar posisi dua peminjam dalam `queue`
 - viii. Buat `processSpecialBooks` untuk memproses buku terkutuk pada peminjam pertama dalam daftar.

2. Source Code

```
Book.java
public class Book {
    private String title;
    private String author;
    private String genre;
    private String status;

    public Book(String title, String author, String genre,
String status) {
        this.title = title;
        this.author = author;
        this.genre = genre;
```

```

        this.status = status;
    }

    public String getStatus() {
        return status;
    }

    public void displayBook() {
        System.out.println("=====
=====");
        System.out.println("Judul Buku\t: " + title);
        System.out.println("Pengarang\t: " + author);
        System.out.println("Genre\t\t: " + genre);
        System.out.println("Status Buku\t: " + status);
    }
}

```

Borrower.java

```

import java.util.Stack;

public class Borrower {
    private String name;
    private boolean specialCard;
    private Stack<Book> borrowedBooks;

    public Borrower(String name, boolean specialCard,
Stack<Book> borrowedBooks) {
        this.name = name;
        this.specialCard = specialCard;
        this.borrowedBooks = borrowedBooks;
    }

    public boolean hasSpecialCard() {
        return specialCard;
    }

    public String getName() {
        return name;
    }

    public void displayBorrower(int queueNumber) {
        System.out.println("=====
=====");
        System.out.println("Nama\t\t: " + name);
        System.out.println("Antrian ke\t: " + queueNumber);
        System.out.println("Jumlah Buku\t: " +
borrowedBooks.size());
        System.out.println("Kartu Spesial\t: " + (specialCard
? "Ada" : "Tidak ada"));
    }

    public void displayBooks() {
        System.out.println("=====
=====");
        System.out.println("=
name.toUpperCase() + "
        =");
        for (Book book : borrowedBooks) {
            book.displayBook();
        }
    }
}

```

```

        public void removeNonSpecialBooks() {
            borrowedBooks.removeIf(book ->
                book.getStatus().equals("Cursed") && !specialCard);
        }
    }

LibraryQueue.java
import java.util.LinkedList;
import java.util.Queue;

public class LibraryQueue {
    private Queue<Borrower> queue;

    public LibraryQueue() {
        queue = new LinkedList<>();
    }

    public void addBorrower(Borrower borrower) {
        queue.add(borrower);
    }

    public void removeFirstBorrower() {
        queue.poll();
    }

    public void showQueue() {
        System.out.println("=====
=====");
        System.out.println("=
ANTRIAN                                =")
        int position = 1;
        for (Borrower borrower : queue) {
            borrower.displayBorrower(position);
            position++;
        }
    }

    public void displayBorrowerBooks(String name) {
        for (Borrower borrower : queue) {
            if (borrower.getName().equalsIgnoreCase(name)) {
                borrower.displayBooks();
                return;
            }
        }
        System.out.println("Peminjam tidak ditemukan dalam
antrian.");
    }

    public void swapQueue() {
        if (queue.size() >= 2) {
            Borrower first = queue.poll();
            Borrower second = queue.poll();
            queue.add(second);
            queue.add(first);
        }
    }

    public void processSpecialBooks() {
        Borrower currentBorrower = queue.peek();
        if (currentBorrower != null) {
            currentBorrower.removeNonSpecialBooks();
        }
    }
}

```

```

        currentBorrower.displayBooks();
    }
}

Main.java
import java.util.Stack;

public class Main {
    public static void main(String[] args) {
        LibraryQueue libraryQueue = new LibraryQueue();

        // Tambah antrian
        System.out.println("\n# Antrian Awal");

        Stack<Book> booksKazuma = new Stack<>();
        booksKazuma.push(new Book("Cara Menjadi Orang Kaya",
            "Teguh", "Fantasi", "Buku Biasa"));
        booksKazuma.push(new Book("Belajar Java", "Raysen",
            "Edukasi", "Buku Biasa"));
        libraryQueue.addBorrower(new Borrower("Kazuma", false,
            booksKazuma));

        Stack<Book> booksHuTao = new Stack<>();
        booksHuTao.push(new Book("Belajar Ilmu Hitam",
            "Megumin", "Unknown", "Cursed"));
        booksHuTao.push(new Book("Belajar C++", "Raysen",
            "Edukasi", "Buku Biasa"));
        libraryQueue.addBorrower(new Borrower("Hu Tao", true,
            booksHuTao));

        Stack<Book> booksKafka = new Stack<>();
        booksKafka.push(new Book("Cara Menjadi Milioner Dalam
            1 Jam", "Optimus", "Edukasi", "Buku Biasa"));
        booksKafka.push(new Book("Misteri Menghilangnya Nasi
            Puyung", "Optimus", "Misteri", "Buku Biasa"));
        booksKafka.push(new Book("Raysen the Forgotten One",
            "Unknown", "Sejarah", "Cursed"));
        libraryQueue.addBorrower(new Borrower("Kafka", false,
            booksKafka));

        Stack<Book> booksXiangling = new Stack<>();
        booksXiangling.push(new Book("Cara Memasak", "Liyue",
            "Keseharian", "Buku Biasa"));
        libraryQueue.addBorrower(new Borrower("Xiangling",
            false, booksXiangling));

        libraryQueue.showQueue();

        libraryQueue.displayBorrowerBooks("Kazuma");

        // Menghapus Kazuma dari antrian
        System.out.println("\n# Kazuma selesai");
        libraryQueue.removeFirstBorrower();
        libraryQueue.showQueue();

        // Menampilkan buku yang dipinjam oleh Hu Tao
        libraryQueue.displayBorrowerBooks("Hu Tao");

        // Menghapus Hu Tao dari antrian
        System.out.println("\n# Hu Tao Selesai");
    }
}

```



```

        libraryQueue.removeFirstBorrower();
        libraryQueue.showQueue();

        // Menambah Sucrose ke dalam antrian
        Stack<Book> booksSucrose = new Stack<>();
        booksSucrose.push(new Book("Durin The Forgotten
Dragon", "Gold", "Misteri", "Buku Biasa"));
        booksSucrose.push(new Book("Alhcemy", "Albedo",
"Sains", "Cursed"));
        booksSucrose.push(new Book("Resurrection", "Unknown",
"Unknown", "Cursed"));
        libraryQueue.addBorrower(new Borrower("Sucrose", true,
booksSucrose));
        libraryQueue.showQueue();

        // Menghapus Xiangling dari antrian
        libraryQueue.removeFirstBorrower();
        libraryQueue.showQueue();

        // Menukar posisi Kafka dengan Sucrose
        System.out.println("\n# Langkah 8: Menukar posisi Kafka
dengan Sucrose");
        libraryQueue.swapQueue();
        libraryQueue.showQueue();

        // Menampilkan buku yang dipinjam oleh Sucrose
        libraryQueue.displayBorrowerBooks("Sucrose");

        // Menghapus Sucrose dari antrian
        System.out.println("\n# Sucrose selesai");
        libraryQueue.removeFirstBorrower();
        libraryQueue.showQueue();

        // Menampilkan buku yang dipinjam oleh Kafka
        libraryQueue.displayBorrowerBooks("Kafka");

        System.out.println("\n# haspus buku terkutuk");
        libraryQueue.processSpecialBooks();

        // Kafka keluar dari antrian
        System.out.println("\n# Kafka keluar");
        libraryQueue.removeFirstBorrower();
        libraryQueue.showQueue();
    }
}

```

1.3 ANALISIS DATA

1.5.1 Easy Mode

```
public class Book {
    private String title;
    private String author;
    private String genre;
    private String status;

    public Book(String title, String author, String genre, String status)
    {
        this.title = title;
        this.author = author;
        this.genre = genre;
        this.status = status;
    }
    public String getStatus() {
        return status;
    }
}
```

Script “class Book” adalah menginisialisasi beberapa variabel *instance* yaitu *title*, *author*, *genre* dan *status*. Kemudian terdapat konstrktor yang menerima parameter atribut buku tersebut. *Script* “getStatus” untuk mengembalikan nilai dari variabel status.

```
public void displayBook() {
    System.out.println("=====");
    System.out.println("Judul Buku\t: " + title);
    System.out.println("Pengarang\t: " + author);
    System.out.println("Genre\t\t: " + genre);
    System.out.println("Status Buku\t: " + status);
}
}
```

Script “public void displayBook” adalah untuk menampilkan informasi lengkap mengenai buku yaitu seperti judul buku, pengarang, genre dan status, semua informasi di tampilkan dengan “\t” agar hasilnya rapi.

```
public class Borrower {
    private String name;
    private boolean specialCard;
    private Stack<Book> borrowedBooks;

    public Borrower(String name, boolean specialCard, Stack<Book>
    borrowedBooks) {
        this.name = name;
        this.specialCard = specialCard;
        this.borrowedBooks = borrowedBooks;
    }

    public boolean hasSpecialCard() {
        return specialCard;
    }

    public String getName() {
        return name;
    }
}
```

Script di atas menginisialisasi atribut nama, *specialcard* dan *borowwerBooks* serta membuat konstruktor untuk mengalokasikan atribut tersebut. *Script* “hasSpecialCard” untuk mengembalikan nilai *specialcard* ketika peminjam memiliki kartu spesial. *Script* “getName” mengembalikan nama peminjam.

```
public void displayBorrower(int queueNumber) {
    System.out.println("=====
=====");
    System.out.println("Nama\t\t: " + name);
    System.out.println("Antrian ke\t: " + queueNumber);
    System.out.println("Jumlah Buku\t: " + borrowedBooks.size());
    System.out.println("Kartu Spesial\t: " + (specialCard ? "Ada" :
    "Tidak ada"));
}

    public void displayBooks() {
        System.out.println("=====
=====");
        System.out.println("=                BUKU " + name.toUpperCase()
+ "
        =");
        for (Book book : borrowedBooks) {
            book.displayBook();
        }
    }

    public void removeNonSpecialBooks() {
        borrowedBooks.removeIf(book -> book.getStatus().equals("Cursed")
        && !specialCard);
    }
}
```

Script “displayBorrower” adalah untuk menampilkan informasi peminjam seperti nama, nomor antrian, jumlah buku dan status. Ambil parameter “queueNumber” sebagai nomor antrian peminjam. *Script* “displayBooks” adalah untuk menampilkan informasi lengkap semua buku yang di pinjam. *Script* “removeSpecialBooks” adalah menghapus buku dengan status cursed dari “borrowedBook” jika peminjam tidak memiliki kartu spesial.

```
import java.util.LinkedList;
import java.util.Queue;

public class LibraryQueue {
    private Queue<Borrower> queue;

    public LibraryQueue() {
        queue = new LinkedList<>();
    }

    public void addBorrower(Borrower borrower) {
        queue.add(borrower);
    }

    public void removeFirstBorrower() {
        queue.poll();
    }
}
```

Script “Queue<Borrower>” adalah objek yang menyimpan daftar peminjam yang sedang dalam antrian. Konstruktor “LibraryQueue” menginisialisasi *queue* sebagai *instace* dari *linkedlist* dengan tipe data *queue*.

Fungsi “addBorrower” adalah untuk menambahkan peminjam ke antrian dan akan di tempatkan di belakang. Fungsi “removeFirstBorrower” adalah untuk menghapus peminjam yang berada pada antrian terdepan.

```
public void showQueue() {
    System.out.println("=====");
    System.out.println("=  DAFTAR ANTRIAN  =");
    int position = 1;
    for (Borrower borrower : queue) {
        borrower.displayBorrower(position);
        position++;
    }
}

public void displayBorrowerBooks(String name) {
    for (Borrower borrower : queue) {
        if (borrower.getName().equalsIgnoreCase(name)) {
            borrower.displayBooks();
            return;
        }
    }
    System.out.println("Peminjam tidak ditemukan dalam antrian.");
}
```

Script “showQueue” adalah fungsi untuk menampilkan seluruh peminjam dalam antrian beserta posisi mereka, menggunakan perulangan *for-each* untuk melewati setiap *borrower* dalam *queue* dan memanggil fungsi “displayBorrower” menampilkan informasi peminjam dengan nomor posisi, variabel *position* digunakan untuk melacak posisi peminjam dalam antrian. Fungsi “displayBook” untuk menampilkan semua buku yang dipinjam oleh peminjam berdasarkan nama yang dicari.

```
public void swapQueue() {
    if (queue.size() >= 2) {
        Borrower first = queue.poll();
        Borrower second = queue.poll();
        queue.add(second);
        queue.add(first);
    }
}

public void processSpecialBooks() {
    Borrower currentBorrower = queue.peek();
}
```

```
        if (currentBorrower != null) {  
            currentBorrower.removeNonSpecialBooks();  
            currentBorrower.displayBooks();  
        }  
    }  
}
```

Fungsi “swapQueue” adalah untuk menukar posisi dua peminjam dalam *queue*, jika terdapat setidaknya dua peminjam, maka fungsi mengambil dua peminjam pertama dengan “poll()”, lalu menambah ke dalam urutan *queue* yang terbalik. Fungsi “prosesSpecialBooks” adalah untuk memproses buku – buku terkutuk (status *cursed*) pada peminjam pertama dalam *queue*. (status *cursed*) pada peminjam pertama dalam *queue*. Jika peminjam pertama memiliki buku dengan status *cursed* dan tidak memiliki kartu spesial, maka buku tersebut akan dihapus dari daftar.