MODUL II
STACK AND QUEUE

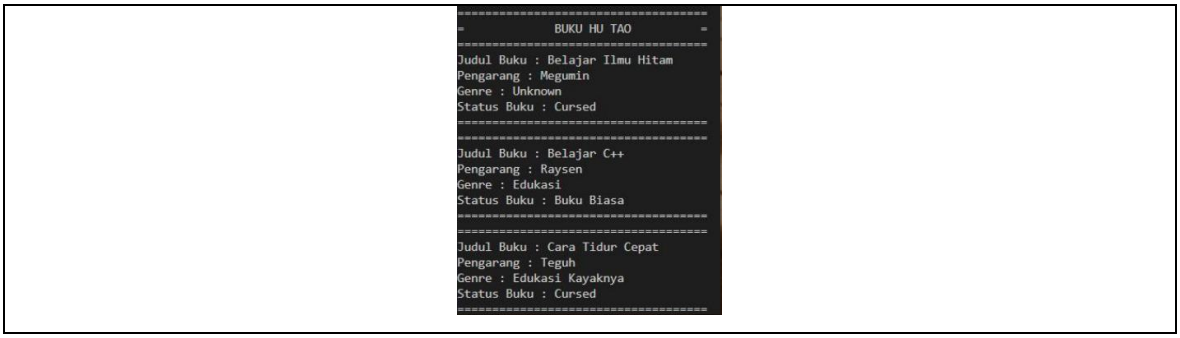
2.1 PERMASALAHAN

2.1.1 Antrian Pemusing Kepala

Dihari berikutnya Yanto Kembali menjadi volunteer penjaga perpustakaan sendirian 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, merekapun berbaris di depan meja Yanto dengan barisan seperti gambar ini.

Kazuma maju dan menunjukkan buku – buku yang ingin dia pinjam kepada Yanto, Kazuma meminjam 2 buku yang berjudul “Belajar Java” dan “Cara Menjadi Orang Kaya”, atribut lengkap bukunya dapat dilihat pada gambar di bawah. Setelah Kazuma menyelesaikan peminjamannya, ia pun pergi. Sekarang tinggal tersisa 3 orang dalam antrian, dan antrian setelah Kazuma menjadi antrian pertama. (Buat antrian menggunaakn Queue dan setiap orang dalam antrian memiliki tumpukan buku (Stack). “Antrian ke” dan “Jumlah Buku” harus dinamis).

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 di bawah



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 pengetahuan – pengetahuan terlarang jadi untuk meminjam buku terkututk, peminjam harus memiliki kartu special. Karena Hu Tao memiliki kartu special, dia dapat meminjam buku terkutuk dan setelah menyelesaikan peminjammanya, Hu Tao langsung pergi. Di antrian sekarang tinggal 2 orang.

Setelah Hu tao keluar, seseorang datang dengan terburu – buru masuk ke dalam antrian. Orang itu adalah Sucrose, ia tampaknya ingin meminjma buku juga. Di tangannya terdapat 3 tumpukan buku.

Xiangling tiba – tiba mendapat panggilan dari Zhongli, Yanto bisa mendengar samar – samar percakapan Xiangling di teleponnya, 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 (Gunakan temp stack untuk menghapus node ditengah stack. Gunakan method swap untuk menukar posisi node dalam queue).

Sucrose menunjukkan buku – buku yang ingin ia pinjam ke Yanto, Sucrose meminjam 3 buku denga njudul “Resurection”, “Alchemy” dan “Durin The Forgotten Dragon”, atribut lengkap bukunya dapat dilihat pada gambar di bawah.

Setelah Sucrose menyelesaikan peminjamannya, ia pun pergi denga terburu – buru. Sekarang tinggal tersisa 1 orang dalam antrian yaitu Kafka.

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

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 punmemberi tahu Kafka bahwa Kafka memerlukan kartu special untuk meminjam buku terkutuk. Yanto pun memindahkan buku yang terkutuk ke bagian paling atas tumpukan dan meminta Kafka untuk mengembalikan buku tersebut. Setelah Kafka mengembalikan buku yang terkutuk, ia menyelesaikan proses peminjaman. Sudah tidak ada orang dalam antrian (Gunakan method swap untuk menukar posisi buku dalam stack).

```
==================================
=           BUKU KAFKA           =
==================================
Judul Buku : Cara Menjadi Milioner Dalam 1 Jam
Pengarang : Master Oogway
Genre : Edukasi
Status Buku : Buku Biasa
==================================
==================================
Judul Buku : Misteri Menghilangnya Nasi Puyung
Pengarang : Optimus
Genre : Misteri
Status Buku : Buku Biasa
==================================
==================================
Judul Buku : Raysen the Forgotten One
Pengarang : Unknown
Genre : Sejarah
Status Buku : Cursed
==================================
Tidak Bisa Meminjam Buku Terkutuk Bila Tidak Memiliki Kartu Spesial
```

```
==================================
=           BUKU KAFKA           =
==================================
Judul Buku : Raysen the Forgotten One
Pengarang : Unknown
Genre : Sejarah
Status Buku : Cursed
==================================
==================================
Judul Buku : Misteri Menghilangnya Nasi Puyung
Pengarang : Optimus
Genre : Misteri
Status Buku : Buku Biasa
==================================
==================================
Judul Buku : Cara Menjadi Milioner Dalam 1 Jam
Pengarang : Master Oogway
Genre : Edukasi
Status Buku : Buku Biasa
==================================
```

```
==================================
=           BUKU KAFKA           =
==================================
Judul Buku : Misteri Menghilangnya Nasi Puyung
Pengarang : Optimus
Genre : Misteri
Status Buku : Buku Biasa
==================================
==================================
Judul Buku : Cara Menjadi Milioner Dalam 1 Jam
Pengarang : Master Oogway
Genre : Edukasi
Status Buku : Buku Biasa
==================================
```

2.2 HASIL PERCOBAAN

2.2.1 Antrian Pemusing Kepala

1. Algoritma
    a. *Start*.
    b. Inisialisasi beberapa objek Stack untuk kazuma, huTao, kafka, xiangLing, dan sucrose sebagai tumpukan buku tiap peminjam.
    c. Tambahkan buku ke kazuma dengan judul "Belajar Java" oleh "Raysen" dan "Cara Menjadi Orang Kaya" oleh "Teguh".
    d. Tambahkan buku ke huTao dengan judul "Cara Tidur Cepat" oleh "Pengarang C", "Belajar C++" oleh "Pengarang D", dan "Belajar Ilmu Hitam" oleh "Pengarang E".
    e. Tambahkan buku ke kafka dengan judul "Raysen the Forgotten One" oleh "Unknown", "Misteri Menghilangnya Nasi Puyung" oleh "Optimus", dan "Cara Menjadi Milioner Dalam 1 Jam" oleh "Master Oogway".
    f. Tambahkan satu buku ke xiangLing berjudul "Misteri Air Limbah Naga" oleh "Unknown".
    g. Tambahkan buku ke sucrose dengan judul "Resurection" oleh "Unknown", "Alchemy" oleh "Albedo", dan "Durin The Forgotten Dragon" oleh "Gold".
    h. Inisialisasi objek Queue bernama Q untuk mengatur antrian peminjam.
    i. Tambahkan kazuma ke antrian dengan nama "Kazuma" tanpa kartu spesial.
    j. Tambahkan kazuma ke antrian dengan nama "Kazuma" tanpa kartu spesial.
    k. Tambahkan huTao ke antrian dengan nama "Hu Tao" dan kartu spesial.
    l. Tambahkan kafka ke antrian dengan nama "Kafka" tanpa kartu spesial.
    m. Tambahkan xiangLing ke antrian dengan nama "XiangLing" tanpa kartu spesial. Tampilkan semua peminjam dan buku mereka dalam antrian.
    n. Hapus peminjam pertama, yaitu Kazuma, dari antrian dan tampilkan buku-buku yang dipinjamnya. Tampilkan kembali antrian setelah Kazuma dihapus.
    o. Hapus peminjam berikutnya, yaitu Hu Tao, dari antrian dan tampilkan buku-buku yang dipinjamnya. Tampilkan antrian setelah Hu Tao dihapus.
    p. Tambahkan sucrose ke antrian dengan nama "SUCROSE" dan kartu spesial. Tampilkan antrian setelah Sucrose ditambahkan.
    q. Hapus XiangLing dari tengah antrian. Tampilkan antrian setelah XiangLing dihapus.
    r. Tukar posisi Kafka dan SUCROSE dalam antrian. Tampilkan antrian setelah swap posisi Kafka dan Sucrose.

s. Hapus peminjam pertama dalam antrian. Tampilkan antrian setelah peminjam pertama dihapus lagi.

t. Tukar posisi buku pertama dan ketiga dalam stack kafka. Tampilkan daftar buku dalam stack kafka setelah swap.

u. Hapus buku teratas dalam stack kafka. Hapus peminjam pertama dalam antrian lagi. Tampilkan kondisi antrian terakhir setelah seluruh operasi selesai.

v. *End*.

2. *Source Code*

```
public class StackNode {
    String judul;
    String pengarang;
    String genre;
    boolean terkutuk;
    StackNode next;

    public StackNode(String judul, String pengarang, String genre,
boolean terkutuk){
        this.judul = judul;
        this.pengarang = pengarang;
        this.genre = genre;
        this.terkutuk = terkutuk;
        this.next = null;
    }
}

public class Stack {
    StackNode top;

    public Stack() {
        top = null;
    }

    public void push(String judul, String pengarang, String genre,
boolean terkutuk) {
        StackNode newNode = new StackNode(judul, pengarang, genre,
terkutuk);
        newNode.next = top;
        top = newNode;
    }
```

```
public int hitung() {
    int pos = 0;
    StackNode temp = top;
    while (temp != null) {
        pos++;
        temp = temp.next;
    }
    return pos;
}

public void pop(){
    if(top == null){
        System.out.println("Kosong");
        return;
    }else{
        top = top.next;
    }
}

public void swap(int pos1, int pos2){
    StackNode prevNode1 = null;
    StackNode prevNode2 = null;
    StackNode currNode1 = top;
    StackNode currNode2 = top;
    for (int i = 1; i<pos1; i++) {
        prevNode1 = currNode1;
        currNode1 = currNode1.next;
    }
    for (int i = 1; i<pos2; i++) {
        prevNode2 = currNode2;
        currNode2 = currNode2.next;
    }
    if(prevNode1 != null){
        prevNode1.next = currNode2;
    }else{
        top = currNode2;
    }
    if(prevNode2 != null){
        prevNode2.next = currNode1;
    }else{
```

```
            top = currNode1;
        }
        StackNode temp = currNode1.next;
        currNode1.next = currNode2.next;
        currNode2.next = temp;
    }

    public void displayBukuStack(String nama) {
        StackNode temp = top;
        System.out.println("====================================");
        System.out.println("=           BUKU "+ nama+"           =");
        System.out.println("====================================");
        while (temp != null) {

System.out.println("====================================");
            System.out.println("Judul   Buku:   "+temp.judul+"\nNama
Pengarang: "+temp.pengarang+"\nGenre: "+temp.genre +"\nStatus Buku:
"+(temp.terkutuk ? "Cursed" : "Buku Biasa"));

System.out.println("====================================");
            temp = temp.next;
        }
    }
}

public class QueueNode {
    String nama;
    int antrian;
    boolean spesial;
    Stack stack;
    QueueNode next;

    public QueueNode(String nama, int antrian, boolean spesial, Stack
stack){
        this.nama = nama;
        this.antrian = antrian;
        this.spesial = spesial;
        this.stack = stack;
        this.next = null;
    }
}
```

```
public class Queue {
    QueueNode head;
    QueueNode tail;
    private int currAntrian;

    public Queue() {
        this.head = null;
        this.tail = null;
        this.currAntrian = 1;
    }


    public void enqueue(String nama, boolean spesial, Stack stack) {
        QueueNode newNode = new QueueNode(nama, currAntrian++,
spesial, stack);
        if (tail == null) {
            head = newNode;
            tail = newNode;
        } else {
            tail.next = newNode;
            tail = newNode;
        }
    }

    public void displayBukuQueue(QueueNode node) {
        if (node == null || node.stack == null) {
            System.out.println("Kosong");
            return;
        }
        StackNode temp = node.stack.top;
        System.out.println("====================================");
        System.out.println("=                 BUKU "+ node.nama +"
=");
        System.out.println("====================================");
        while (temp != null) {

System.out.println("====================================");
            System.out.println("Judul Buku: "+temp.judul +"\nNama
Pengarang: "+temp.pengarang+"\nGenre: "+temp.genre +"\nStatus Buku:
"+(temp.terkutuk ? "Cursed" : "Buku Biasa"));
```

```
System.out.println("==================================");
            temp = temp.next;
        }
    }

    public void dequeue() {
        if (head == null) {
            System.out.println("Antrian Kosong");
            return;
        }
        QueueNode temp = head;
        boolean adaTerkutuk = false;
        StackNode stackNode = temp.stack.top;
        while (stackNode != null) {
            if (stackNode.terkutuk) {
                adaTerkutuk = true;
                break;
            }
            stackNode = stackNode.next;
        }
        if (adaTerkutuk && !temp.spesial) {
            displayBukuQueue(temp);
            System.out.println("Tidak Bisa Meminjam Buku Terkutuk
Bila Tidak Memiliki Kartu Spesial");
            return;
        }
        head = head.next;
        if (head == null) {
            tail = null;
        }
        displayBukuQueue(temp);
        QueueNode current = head;
        int newAntrian = 1;
        while (current != null) {
            current.antrian = newAntrian++;
            current = current.next;
        }
        currAntrian = newAntrian;
    }
```

```
    public void deleteMid(String nama){
        QueueNode current = head;
        QueueNode prev = null;
        while (current != null && !current.nama.equals(nama)) {
            prev = current;
            current = current.next;
        }
        prev.next = current.next;
        int newAntrian = 1;
        while (current != null) {
            current.antrian = newAntrian++;
            current = current.next;
        }
        currAntrian = newAntrian;
    }

    public void swapAntrian(String nama1, String nama2) {
        QueueNode prevNode1 = null;
        QueueNode node1 = head;
        while (node1 != null && !node1.nama.equals(nama1)) {
            prevNode1 = node1;
            node1 = node1.next;
        }
        QueueNode prevNode2 = null;
        QueueNode node2 = head;
        while (node2 != null && !node2.nama.equals(nama2)) {
            prevNode2 = node2;
            node2 = node2.next;
        }
        if(prevNode1 !=null){
            prevNode1.next = node2;
        }else{
            head = node2;
        }
        if(prevNode2 !=null){
            prevNode2.next = node1;
        }else{
            head = node1;
        }
        QueueNode temp = node1.next;
        node1.next = node2.next;
```

```
        node2.next = temp;
        QueueNode current = head;
        int newAntrian = 1;
        while (current != null) {
            current.antrian = newAntrian++;
            current = current.next;
        }
        currAntrian = newAntrian;
    }

    public void displayAntrian() {
        QueueNode temp = head;
        System.out.println("=====================================");
        System.out.println("=          DAFTAR ANTRIAN           =");
        System.out.println("=====================================");
        if(temp == null){
            System.out.println("Antrian Kosong");
            return;
        }
        while (temp != null) {
            int jumlahBuku = temp.stack != null ? temp.stack.hitung()
: 0;

System.out.println("=====================================");
            System.out.println("Nama: " + temp.nama + "\nAntrian ke
: "+temp.antrian+"\nJumlah Buku : "+jumlahBuku +"\nKartu Spesial :
"+ (temp.spesial ? "ada" : "tidak ada"));

System.out.println("=====================================");
            temp = temp.next;
        }
    }
}

public class Hardd {
    public static void main(String[] args) {
        Stack kazuma = new Stack();
        kazuma.push("Belajar Java", "Raysen", "Edukasi", false);
        kazuma.push("Cara Menjadi Orang Kaya", "Teguh", "Fantasi",
false);
```

```
        Stack huTao = new Stack();
        huTao.push("Cara Tidur Cepat", "Pengarang C", "Kesehatan",
true);
        huTao.push("Belajar C++", "Pengarang D", "Pendidikan",
false);
        huTao.push("Belajar Ilmu Hitam", "Pengarang E", "Terlarang",
true);

        Stack kafka = new Stack();
        kafka.push("Raysen the Forgotten One", "Unknown", "Sejarah",
true);
        kafka.push("Misteri Menghilangnya Nasi Puyung", "Optimus",
"Misteri", false);
        kafka.push("Cara Menjadi Milioner Dalam 1 Jam", "Master
Oogway", "Edukasi", false);

        Stack xiangLing = new Stack();
        xiangLing.push("Misteri Air Limbah Naga", "Unknown",
"Horror", false);

        Stack sucrose = new Stack();
        sucrose.push("Resurection", "Unknown", "Unknown", true);
        sucrose.push("Alhcemy", "Albedo", "Sience", true);
        sucrose.push("Durin The Forgotten Dragon", "Gold", "Misteri",
false);

        Queue Q = new Queue();
        Q.enqueue("Kazuma", false, kazuma);
        Q.enqueue("Hu Tao", true, huTao);
        Q.enqueue("Kafka", false, kafka);
        Q.enqueue("XiangLing", false, xiangLing);

        Q.displayAntrian();
        Q.dequeue();
        Q.displayAntrian();
        Q.dequeue();
        Q.displayAntrian();
        Q.enqueue("SUCROSE", true, sucrose);
        Q.displayAntrian();
        Q.deleteMid("XiangLing");
        Q.displayAntrian();
```

```
        Q.swapAntrian("Kafka", "SUCROSE");
        Q.displayAntrian();
        Q.dequeue();
        Q.displayAntrian();
        Q.dequeue();
        kafka.swap(1, 3);
        kafka.displayBukuStack("Kafka");
        kafka.pop();
        Q.dequeue();
        Q.displayAntrian();
    }
}
```

2.3 ANALISIS DATA

2.3.1 Nama Program

```
public class StackNode {
    String judul;
    String pengarang;
    String genre;
    boolean terkutuk;
    StackNode next;

    public StackNode(String judul, String pengarang, String genre,
boolean terkutuk){
        this.judul = judul;
        this.pengarang = pengarang;
        this.genre = genre;
        this.terkutuk = terkutuk;
        this.next = null;
    }
}
```

Class berikut berfungsi sebagai elemen untuk menyimpan data terkait stack. Setiap node menyimpan data tentang judul, pengarang, genre, dan status "terkutuk", serta referensi ke node berikutnya, memungkinkan untuk membentuk struktur data linked list untuk mengelola item dalam urutan tertentu.

```
public class Stack {
    StackNode top;

    public Stack() {
        top = null;
    }
```

Class berikut mereferensikan node teratas dalam stacknya. Jika "`top`" adalah null, itu artinya stacknya kosong. Lalu "`top`" di set ke null, untuk menandakan bahwa stack tidak memiliki elemen.

```
public void push(String judul, String pengarang, String genre, boolean
terkutuk) {
        StackNode newNode = new StackNode(judul, pengarang, genre,
terkutuk);
        newNode.next = top;
        top = newNode;
    }
```

Class ini berfungsi sebagai menambahkan elemen baru ke stack nya. Membuat objek "`StackNode`" baru, mengatur "`next`" dari node baru untuk menunjuk ke node teratas sebelumnya, lalu memperbarui "`top`" untuk menunjuk ke node baru, sehingga node baru menjadi node teratas.

```
public int hitung() {
        int pos = 0;
        StackNode temp = top;
        while (temp != null) {
            pos++;
```

```
            temp = temp.next;
        }
        return pos;
    }
```

Class ini berfungsi sebagai menghitung jumlah elemen dalam stack nya. Menginisialisasi variabel "pos" untuk menghitung posisinya, melakukan looping melalui stacknya untuk menunjuk ke node saat ini, untuk setiap node yang ditemukan akan ditambah satu ke "pos" sampai mencapai akhir dari stack.

```
public void pop(){
        if(top == null){
            System.out.println("Kosong");
            return;
        }else{
            top = top.next;
        }
    }
```

Class ini digunakan untuk menghapus elemen teratas dari stack. Class ini awalnya akan memeriksa apakah "top" nya kosong, jika kosong, maka program akan menampilkan "Kosong", jika ada isinya, maka akan mengatur "top" untuk menunjuk ke node berikutnya, sehingga node teratas yang sebelumnya dihapus dari stack.

```
public void swap(int pos1, int pos2){
        StackNode prevNode1 = null;
        StackNode prevNode2 = null;
        StackNode currNode1 = top;
        StackNode currNode2 = top;
        for (int i = 1; i<pos1; i++) {
            prevNode1 = currNode1;
            currNode1 = currNode1.next;
        }
        for (int i = 1; i<pos2; i++) {
            prevNode2 = currNode2;
            currNode2 = currNode2.next;
        }
        if(prevNode1 != null){
            prevNode1.next = currNode2;
        }else{
            top = currNode2;
        }
        if(prevNode2 != null){
            prevNode2.next = currNode1;
        }else{
            top = currNode1;
        }
        StackNode temp = currNode1.next;
        currNode1.next = currNode2.next;
        currNode2.next = temp;
    }
```

Class ini memiliki tujuan untuk menukan posisi kedua node yang ada dalam stack berdasarkan posisi mereka, bisa dilakukan dengan meng input posisinya seperti "pos1" ke "pos2".

```
public void displayBukuStack(String nama) {
        StackNode temp = top;
        System.out.println("==================================");
        System.out.println("=           BUKU "+ nama+"           =");
        System.out.println("==================================");
        while (temp != null) {
            System.out.println("==================================");
            System.out.println("Judul     Buku:     "+temp.judul+"\nNama
Pengarang:  "+temp.pengarang+"\nGenre:  "+temp.genre  +"\nStatus  Buku:
"+(temp.terkutuk ? "Cursed" : "Buku Biasa"));
            System.out.println("==================================");
            temp = temp.next;
        }
    }
}
```

Class ini bertujuan untuk menampilkan semua buku yang ada dalam stacknya dengan format yang ada di kodingan berikut.

```
public class QueueNode {
    String nama;
    int antrian;
    boolean spesial;
    Stack stack;
    QueueNode next;

     public QueueNode(String nama, int antrian, boolean spesial, Stack
stack){
        this.nama = nama;
        this.antrian = antrian;
        this.spesial = spesial;
        this.stack = stack;
        this.next = null;
    }
}
```

Class ini digunakan untuk menyimpan data dalam antrian atau queue, menangani elemen special dengan variabel "`spesial`", lalu akan mengelola stack dengan adanya referensi "`stack`".

```
public class Queue {
    QueueNode head;
    QueueNode tail;
    private int currAntrian;

    public Queue() {
        this.head = null;
        this.tail = null;
        this.currAntrian = 1;
    }
```

Class ini merupakan sebuah penginisialisasian pada head, tail, currAntrian. Lalu konstruktor digunakan untuk menginisialisasi antrian baru, dengan "`head`" dan "`tail`" di set ke null untuk menandakan antriannya kosong, lalu "`currAntrian`" diatur ke 1 untuk penomoran awal untuk elemen baru yang nantinya akan ditambahkan ke dalam antrian berikut.

```
public void enqueue(String nama, boolean spesial, Stack stack) {
        QueueNode newNode = new QueueNode(nama, currAntrian++, spesial,
stack);
        if (tail == null) {
            head = newNode;
            tail = newNode;
        } else {
            tail.next = newNode;
            tail = newNode;
        }
    }
```

Class berikut berfungsi untuk menambahkan elemen baru pada antrian. Membuat objek "`QueueNode`" baru dengan nama, nomor antrian, status special, dan stack, jika "`tail`" nya kosong, ini akan melakukan set "`head`" dan "`tail`" nya ke "`newNode`", jika tidak kosong, "`newNode`" akan ditambahkan diakhir antrian.

```
public void displayBukuQueue(QueueNode node) {
        if (node == null || node.stack == null) {
            System.out.println("Kosong");
            return;
        }
        StackNode temp = node.stack.top;
        System.out.println("====================================");
        System.out.println("=          BUKU "+ node.nama +"          =");
        System.out.println("====================================");
        while (temp != null) {
            System.out.println("====================================");
            System.out.println("Judul   Buku:   "+temp.judul   +"\nNama
Pengarang:  "+temp.pengarang+"\nGenre:  "+temp.genre  +"\nStatus  Buku:
"+(temp.terkutuk ? "Cursed" : "Buku Biasa"));
            System.out.println("====================================");
            temp = temp.next;
        }
    }
```

Class ini memiliki tujuan untuk menampilkan informasi-informasi dari buku dalam antriannya, program akan memeriksa apakah node nya kosong, jika kosong maka akan menampilkan pesan "`Kosong`", jika tidak kosong maka akan melakukan looping melalui stack yang ada di node dan menampilkan judul, pengarang, genre, dan statusnya.

```
public void dequeue() {
        if (head == null) {
            System.out.println("Antrian Kosong");
            return;
        }
        QueueNode temp = head;
        boolean adaTerkutuk = false;
        StackNode stackNode = temp.stack.top;
        while (stackNode != null) {
            if (stackNode.terkutuk) {
                adaTerkutuk = true;
                break;
            }
            stackNode = stackNode.next;
        }
        if (adaTerkutuk && !temp.spesial) {
```

```
            displayBukuQueue(temp);
            System.out.println("Tidak Bisa Meminjam Buku Terkutuk Bila
Tidak Memiliki Kartu Spesial");
            return;
        }
        head = head.next;
        if (head == null) {
            tail = null;
        }
        displayBukuQueue(temp);
        QueueNode current = head;
        int newAntrian = 1;
        while (current != null) {
            current.antrian = newAntrian++;
            current = current.next;
        }
        currAntrian = newAntrian;
    }
```

Class ini bertujuan untuk menghapus elemen terdepan pada antriannya. Program akan memeriksa apakah ada isinya atau tidak, jika tidak maka akan menampilkan pesan "`Antrian Kosong`", jika tidak kosong maka program akan memeriksa apakah ada buku yang terkutuk dalam stack, jika ada dan node tidak memiliki kartu spesial, maka program akan menampilkan buku dan pesan bahwa buku tersebut terkutur dan tidak dapat dipinjam tanpa kartu spesial, jika tidak ada masalah, maka program akan melakukan penghapusan dari antrian terdepan.

```
public void deleteMid(String nama){
        QueueNode current = head;
        QueueNode prev = null;
        while (current != null && !current.nama.equals(nama)) {
            prev = current;
            current = current.next;
        }
        prev.next = current.next;
        int newAntrian = 1;
        while (current != null) {
            current.antrian = newAntrian++;
            current = current.next;
        }
        currAntrian = newAntrian;
    }
```

Class berikut bertujuan untuk menghapus node dari antrian berdasarkan namanya. Melakukan looping untuk menemukan nama yang diberikan, jika ketemu maka program akan menghapusnya dan memperbarui nomor pada antrian untuk list yang tersisa setelah melakukan penghapusan.

```
public void swapAntrian(String nama1, String nama2) {
        QueueNode prevNode1 = null;
        QueueNode node1 = head;
        while (node1 != null && !node1.nama.equals(nama1)) {
            prevNode1 = node1;
            node1 = node1.next;
```

```
        }
        QueueNode prevNode2 = null;
        QueueNode node2 = head;
        while (node2 != null && !node2.nama.equals(nama2)) {
            prevNode2 = node2;
            node2 = node2.next;
        }
        if(prevNode1 !=null){
            prevNode1.next = node2;
        }else{
            head = node2;
        }
        if(prevNode2 !=null){
            prevNode2.next = node1;
        }else{
            head = node1;
        }
        QueueNode temp = node1.next;
        node1.next = node2.next;
        node2.next = temp;
        QueueNode current = head;
        int newAntrian = 1;
        while (current != null) {
            current.antrian = newAntrian++;
            current = current.next;
        }
        currAntrian = newAntrian;
    }
```

Class ini bertujuan untuk menukarkan posisi dari kedua list pada antrian berdasarkan namanya. Setelah menemukan nama dari kedua listnya, program akan mengatur "`next`" dari list sebelumnya untuk menunjuk list yang akan ditukar, lalu setelah melakukan penukaran, program akan memperbarui nomor antriannya.

```
public void displayAntrian() {
        QueueNode temp = head;
        System.out.println("==================================");
        System.out.println("=          DAFTAR ANTRIAN        =");
        System.out.println("==================================");
        if(temp == null){
            System.out.println("Antrian Kosong");
            return;}
        while (temp != null) {
            int jumlahBuku = temp.stack != null ? temp.stack.hitung() :
0;
            System.out.println("==================================");
            System.out.println("Nama: " + temp.nama + "\nAntrian ke :
"+temp.antrian+"\nJumlah Buku : "+jumlahBuku +"\nKartu Spesial : "+
(temp.spesial ? "ada" : "tidak ada"));
            System.out.println("==================================");
            temp = temp.next;
        }
    }
}
```

Class ini bertujuan sebagai menampilkan daftar pada antrian pada saat ini. Memeriksa apakah listnya kosong, jika tidak maka akan mencetak atau menampilkan semua informasi dari list tersebut.

```
public class Hardd {
    public static void main(String[] args) {
        Stack kazuma = new Stack();
        kazuma.push("Belajar Java", "Raysen", "Edukasi", false);
        kazuma.push("Cara Menjadi Orang Kaya", "Teguh", "Fantasi",
false);
        Stack huTao = new Stack();
        huTao.push("Cara Tidur Cepat", "Pengarang C", "Kesehatan",
true);
        huTao.push("Belajar C++", "Pengarang D", "Pendidikan", false);
        huTao.push("Belajar Ilmu Hitam", "Pengarang E", "Terlarang",
true);
        Stack kafka = new Stack();
        kafka.push("Raysen the Forgotten One", "Unknown", "Sejarah",
true);
        kafka.push("Misteri Menghilangnya Nasi Puyung", "Optimus",
"Misteri", false);
        kafka.push("Cara Menjadi Milioner Dalam 1 Jam", "Master Oogway",
"Edukasi", false);
        Stack xiangLing = new Stack();
        xiangLing.push("Misteri Air Limbah Naga", "Unknown", "Horror",
false);
        Stack sucrose = new Stack();
        sucrose.push("Resurection", "Unknown", "Unknown", true);
        sucrose.push("Alhcemy", "Albedo", "Sience", true);
        sucrose.push("Durin The Forgotten Dragon", "Gold", "Misteri",
false);
        Queue Q = new Queue();
        Q.enqueue("Kazuma", false, kazuma);
        Q.enqueue("Hu Tao", true, huTao);
        Q.enqueue("Kafka", false, kafka);
        Q.enqueue("XiangLing", false, xiangLing);
        Q.displayAntrian();
        Q.dequeue();
        Q.displayAntrian();
        Q.dequeue();
        Q.displayAntrian();
        Q.enqueue("SUCROSE", true, sucrose);
        Q.displayAntrian();
        Q.deleteMid("XiangLing");
        Q.displayAntrian();
        Q.swapAntrian("Kafka", "SUCROSE");
        Q.displayAntrian();
        Q.dequeue();
        Q.displayAntrian();
        Q.dequeue();
        kafka.swap(1, 3);
        kafka.displayBukuStack("Kafka");
        kafka.pop();
        Q.dequeue();
        Q.displayAntrian();}}
```

Class ini berfungsi sebagai main program pada semua kodingan sebelumnya. Kode ini membuat system dimana pengguna dapat meminjam buku, memanipulasi antrian peminjaman, dan melakukan berbagai operasi pada koleksi buku mereka, lalu menampilkan semuanya menggunakan fungsi "`displayAntrian()`".