MODUL II

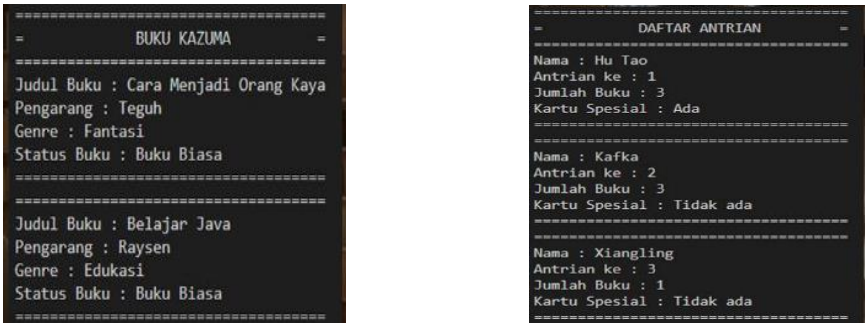
STACK DAN QUEUE

1. PERMASALAHAN

Di hari 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, mereka pun berbaris di depan meja Yanto dengan barisan seperti gambar di bawah.

Gambar 2.1 Daftar Antrian

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 buku-nya dapat di lihat 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. *Note* : Buat antrian menggunakan *Queue* dan setiap orang dalam antrian memiliki tumpukan buku (*Stack*). *Note* : "`Antrian Ke`" dan "`Jumlah Buku`" harus dinamis.



Gambar 2.2 Buku Pinjaman Kazuma

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 buku-nya dapat di lihat pada gambar di bawah.

Gambar 2.3 Buku Pinjaman Hu Tao

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 terkutuk, peminjam harus memiliki kartu *special*. Karena Hu Tao memiliki kartu spesial, dia dapat meminjam buku terkutuk. Setelah menyelesaikan peminjaman-nya, Hu Tao langsung pergi. Di antrian sekarang tertinggal 2 orang.

Gambar 2.4 Antrian Setelah Hu Tao Keluar

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.

Gambar 2.5 Antrian Setelah Sucrose Masuk

Xiangling tiba-tiba mendapat panggilan dari Zhongli, Yanto bisa mendengar samar-samar percakapan xiangling di telepon-nya, Yanto mendengar bahwa Gouba sedang mengamuk di Liyue dan butuh bantuan Xiangling secepat mungkin. Karena urusan

mendesak itu Xiangling menyimpan kembali buku-nya dan langsung pergi. Tinggal tersisa dua orang di antrian.

Gambar 2.6 Antrian Setelah Xiangling Keluar

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. *Note* : Gunakan *temp Stack* untuk menghapus *node* di tengah *Stack*. *Note* : Gunakan *Method Swap* untuk menukar posisi *node* dalam *Queue*.

Gambar 2.7 Antrian Setelah *Swap*

Sucrose menunjukkan buku-buku yang ingin ia pinjam ke Yanto, Sucrose meminjam 3 buku dengan judul "`Resurection`", "`Alhcemy`" dan "`Durin The Forgotten Dragon`", atribut lengkap bukunya dapat dilihat pada gambar di bawah.

Gambar 2.8 Pinjaman Sucrose

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

Gambar 2.9 Antrian 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 mengembalikkan buku tersebut. Setelah Kafka mengembalikan buku yang terkutuk, ia menyelesaikan proses peminjaman. Sudah tidak ada orang dalam antrian.

Gambar 2.10 Pinjaman Kafka

Kondisi setelah *node* di hapus :

Note : Gunakan *method swap* untuk menukar posisi buku dalam *stack*

Gambar 2.11 Pinjaman Kafka Setelah di Hapus

Gambar 2.12 Pinjaman Kafka Setelah di Hapus

2. HASIL PERCOBAAN

2.1 Program

1. Algoritma
   a. Membuat kelas "`Buku`" untuk menyimpan informasi buku yang meliputi *tittle*, *author*, genre, dan status.
   b. Buat kelas "`Stack`" untuk menyimpan data buku sebagai tumpukan (*stack*) dengan metode seperti : metode "`push`", "`pop`", "`special`", "`tukar`", dan "`tampilan`".
   c. Membuat kelas "`Peminjam`" untuk menyimpan data peminjam, yang terdiri dari nama, jenis kartu, antrian, dan jumlah buku yang dipinjam.
   d. Membuat kelas "`Queue`" yang mengelola daftar peminjam menggunakan metode: "`enQueue`", "`deQueue`", "`addBook`", "`tukar`", "`tampilanBuku`", dan "`tampilanAntrian`".
   e. Menambahkan "`Peminjam`" dengan "`enQueue`" dan menentukan kartu "`Peminjam`".
   f. Menambahkan buku yang dipinjam ke "`Peminjam`" terakhir menggunakan "`addBook`". Buku yang ditambahkan masuk ke dalam tumpukan (*stack*) buku milik "`Peminjam`".
   g. Mencetak daftar buku "`Peminjam`" terdepan menggunakan "`TampilanBuku`" dan menghapus "`Peminjam`" dari antrian dengan "`deQueue`" yang memeriksa kartu "`Peminjam`" dan status buku dalam tumpukan. Jika kartu tidak ada dan buku "`Terlarang`" ditemukan, "`Peminjam`" ditolak. Jika "`Peminjam`" memenuhi syarat maka dihapus dari antrian.
   h. Mencetak daftar antrian "`Peminjam`" dan memeriksa jumlah buku yang dipinjam.
   i. Menambahkan "`Peminjam`" baru dan buku baru sesuai kebutuhan.
   j. Menggunakan "`hapus`" untuk menghapus peminjam dari antrian jika diperlukan.
   k. Menggunakan "`tukar`" untuk menukar posisi peminjam dalam antrian dan mencetak daftar buku saat ini dengan "`tampilanBuku`"
   l. Menggunakan "`pop`" untuk menghapus buku dari tumpukan peminjam pertama dan mencetak daftar buku saat ini yang telah diperbaharui.
   m. Menghapus "`Peminjam`" dengan "`deQueue`" hingga antrian kosong.
   n. Mencetak daftar antrian yang tersisa di setiap iterasi untuk memverifikasi pengelolaan antrian.
2. *Source Code*

```
public class Buku {
    public Buku next;
```

```
    public String tittle, author, genre, status;


    public Buku (String tittle, String author, String genre, String
status){
        this.tittle=tittle;
        this.author=author;
        this.genre=genre;
        this.status=status;
    }}

public class Stack {
    public Buku top;

      public void push(String tittle, String author, String genre,
String status){
        Buku dataBaru = new Buku(tittle, author, genre, status);
        if(top==null){
            top=dataBaru;
        }
        else{
            dataBaru.next=top;
            top=dataBaru;
        }}

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

    public boolean spesial(){
        Buku current=top;
        while(current!=null){
            if(current.status=="Terlarang"){
                return true;
            }
            current=current.next;
        }
        return false;
    }

    public void tukar(String tittle1, String tittle2){
        if(tittle1==tittle2){
            System.out.println("Tidak perlu ditukar");
            return;
        }

        Buku prevP=null, currentP=top;
            while(currentP!=null){
                if(currentP.tittle==tittle1){
                    break;
                }
                prevP=currentP;
                currentP=currentP.next;
        }

        Buku prevQ=null, currentQ=top;
            while(currentQ!=null){
```

```
                if(currentQ.tittle==tittle2){
                    break;
                }
                prevQ=currentQ;
                currentQ=currentQ.next;
        }

        if(prevP!=null){
            prevP.next=currentQ;
        }else{
            top=currentQ;
        }

        if(prevQ!=null){
            prevQ.next=currentP;
        }else{
            top=currentP;
        }

        Buku bantu=currentP.next;
        currentP.next=currentQ.next;
        currentQ.next=bantu;
    }

    public void tampilan(){
        Buku current=top;
        while(current!=null){
             System.out.println("==================================
======");
            System.out.println("Judul Buku : " + current.tittle);
            System.out.println("Pengarang : " + current.author);
            System.out.println("Genre : " + current.genre);
            System.out.println("Status Buku : " + current.status);
             System.out.println("==================================
======");
            current=current.next;
        }}}

public class Peminjam {
    public Peminjam next;
    public String nama, kartu;
    public int antrian;
    public int jumlah=0;

    Peminjam(String nama, String kartu){
        this.nama=nama;
        this.kartu=kartu;
    }

    Stack stack = new Stack();
}

public class Queue {
    public Peminjam first;
    public Peminjam last;

    public void enQueue(String nama, String kartu){
        Peminjam dataBaru = new Peminjam(nama, kartu);
        if(first==null){
            first=dataBaru;
            last=dataBaru;
```

```
        }
        else{
            last.next=dataBaru;
            last=dataBaru;
        }}

    public void deQueue(){
        if(first==null){
            System.out.println("Data Kosong");
            return;
        }
        else{
                                    if(first.kartu=="Tidak    Ada"    &&
first.stack.spesial()==true){
                System.out.println("Tidak dapat meminjam buku");
            }
            else{
                first=first.next;
            }}}

    public void addBook(String tittle, String author, String genre,
String status){
        last.stack.push(tittle, author, genre, status);
        last.jumlah++;
    }

    public void hapus(String nama){
        Peminjam current=first;
        while(current!=null){
            if(current.next.nama==nama){
                current.next=current.next.next;
                return;
            }
            current=current.next;
        }
   }

    public void tukar(String nama1, String nama2){
        if(nama1==nama2){
            System.out.println("Tidak perlu ditukar");
            return;
        }

        Peminjam prevP=null, currentP=first;
        while(currentP!=null){
            if(currentP.nama==nama1){
                break;
            }
            prevP=currentP;
            currentP=currentP.next;
        }

        Peminjam prevQ=null, currentQ=first;
        while(currentQ!=null){
            if(currentQ.nama==nama2){
                break;
            }
            prevQ=currentQ;
            currentQ=currentQ.next;
        }
```

```
        if(prevP!=null){
            prevP.next=currentQ;
        }else{
            first=currentQ;
        }

        if(prevQ!=null){
            prevQ.next=currentP;
        }else{
            first=currentP;
        }

        Peminjam bantu=currentP.next;
        currentP.next=currentQ.next;
        currentQ.next=bantu;
    }

    public void tampilanBuku(){
         System.out.println("=======================================
==");
        System.out.println("=\t\tBUKU " + first.nama.toUpperCase() +
"\t\t=");
        first.stack.tampilan();
    }

    public void tampilanAntrian(){
        Peminjam current=first;
        int counter=0;
         System.out.println("=======================================
==");
       System.out.println("=            DAFTAR ANTIAN            =");
        if(current==null){
            System.out.println("Antrian Kosong");
        }
        while(current!=null){
            current.antrian=++counter;
             System.out.println("=================================
======");
            System.out.println("Nama : "+ current.nama);
            System.out.println("Antrian ke : "+ current.antrian);
            System.out.println("Jumlah Buku : "+ current.jumlah);
            System.out.println("Kartu Special : "+ current.kartu);
             System.out.println("=================================
======");
            current=current.next;
        }}}

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

        queue.enQueue("Kazuma", "Tidak Ada");
          queue.addBook("Belajar Java", "Raysen", "Edukasi", "Buku
Biasa");
        queue.addBook("Cara Menjadi Orang Kaya", "Teguh", "Fantasi",
"Buku Biasa");

        queue.enQueue("Hu Tao", "Ada");
       queue.addBook("Cara Tidur Cepat", "Teguh", "Edukasi Kayaknya",
"Terlarang");
```

```
            queue.addBook("Belajar C++", "Raysen", "Edukasi", "Buku
Biasa");
          queue.addBook("Belajar Ilmu Hitam", "Mengumin", "Unknown",
"Terlarang");

         queue.enQueue("Kafka", "Tidak Ada");
             queue.addBook("Raysen the Forgotten One", "Unknown",
"Sejarah", "Terlarang");
        queue.addBook("Misteri Menghilangnya Nasi Puyung", "Optimus",
"Misteri", "Buku Biasa");
          queue.addBook("Cara Menjadi Milioner Dalam 1 Jam", "Master
Oogway", "Edukasi", "Buku Biasa");

         queue.enQueue("Xiangling", "Tidak Ada");
        queue.addBook("Unknown", "Unknown", "Unknown", "Buku Biasa");

         queue.tampilanBuku();
         queue.deQueue();
         queue.tampilanAntrian();
         System.out.println("\n");

         queue.tampilanBuku();
         queue.deQueue();
         queue.tampilanAntrian();
         System.out.println("\n");

         queue.enQueue("Sucrose", "Ada");
                queue.addBook("Resurection",  "Unknown",  "Unknown",
"Terlarang");
         queue.addBook("Alhcemy", "Albedo", "Sience", "Terlarang");
        queue.addBook("Durin The Forgotten Dragon", "Gold", "Misteri",
"Buku Biasa");

         queue.tampilanAntrian();
         System.out.println("\n");

         queue.hapus("Xiangling");
         queue.tampilanAntrian();
         System.out.println("\n");

         queue.tukar("Sucrose", "Kafka");
         queue.tampilanAntrian();
         System.out.println("\n");

         queue.tampilanBuku();
         queue.deQueue();
         queue.tampilanAntrian();
         System.out.println("\n");

         queue.tampilanBuku();
         queue.deQueue();
         System.out.println("\n");

         queue.first.stack.pop();
         queue.tampilanBuku();
         System.out.println("\n");

         queue.deQueue();
         queue.tampilanAntrian();
         System.out.println("\n");
    }}
```

3. ANALISIS DATA

3.1 Program Antrian Pemusing Kepala

```
public class Buku {
public Buku next;
public String tittle, author, genre, status;

public Buku (String tittle, String author, String genre, String status){
       this.tittle=tittle;
       this.author=author;
       this.genre=genre;
       this.status=status;
}}
```

Code di atas merupakan kelas "`Buku`" yang merepresentasikan sebuah objek buku. Kelas "`Buku`" memiliki empat atribut berupa "`tittle`" (judul buku), "`author`" (pengarang), "`genre`" (genre buku), dan "`status`" (status peminjaman buku). Selain itu, terdapat atribut "`next`" yang berfungsi untuk menunjuk ke objek "`Buku`" berikutnya. Konstruktor kelas "`Buku`" digunakan untuk menginisialisasi atribut-atribut buku dengan nilai yang diberikan saat objek dibuat. Sehingga kelas "`Buku`" dapat digunakan untuk membuat dan mengelola koleksi buku dengan efisien.

```
public class Peminjam {
    public Peminjam next;
    public String nama, kartu;
    public int antrian;
    public int jumlah=0;

    Peminjam(String nama, String kartu){
        this.nama=nama;
        this.kartu=kartu;
    }

    Stack stack = new Stack();
}
```

Code di atas merupakan kelas "`Peminjam`" yang merepresentasikan sebuah objek peminjam, di mana kelas ini memiliki atribut "`nama`" (nama peminjam), "`kartu`" (nomor kartu peminjam), "`antrian`" (posisi dalam antrean), dan "`jumlah`" (jumlah buku yang dipinjam dengan inisialisasi awal 0). Atribut "`next`" digunakan untuk menunjuk ke objek "`Peminjam`" berikutnya, yang memungkinkan pengelolaan peminjam dalam struktur data *linked list*. Konstruktor di kelas ini menginisialisasi atribut "`nama`" dan "`kartu`" dengan nilai yang diberikan saat objek dibuat. Objek "`Stack`" yang dideklarasikan untuk menyimpan koleksi buku yang sedang dipinjam oleh peminjam

```
public class Stack {
    public Buku top;
```

```
     public void push(String tittle, String author, String genre, String
status){
        Buku dataBaru = new Buku(tittle, author, genre, status);
        if(top==null){
            top=dataBaru;
        }
        else{
            dataBaru.next=top;
            top=dataBaru;
        }}

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

Script di atas mendefinisikan kelas "`Stack`" yang mengimplementasikan struktur data *stack* (tumpukan) untuk objek"`Buku`". Kelas ini memiliki atribut berupa "`pop`" yang menunjuk ke buku paling atas dalam *stack*. *Method* "`push`" digunakan untuk menambahkan buku baru ke dalam *stack* dengan membuat objek "`Buku`" baru berdasarkan parameter yang diberikan yaitu *tittle*, *author*, genre, dan status. Jika *stack* kosong ("`top`" bernilai *null*), buku baru akan menjadi"`top`". Jika tidak, buku baru akan ditempatkan di atas buku yang sudah ada, dengan mengatur "`next`" dari buku baru ke buku yang sebelumnya ada di "`top`", kemudian memperbarui "`top`" untuk menunjuk ke buku baru. *Method* "`top`" digunakan untuk menghapus buku teratas dari *stack*. Jika *stack* kosong maka akan menampilkan pesan "`Data Kosong`" dan jika tidak maka "`top`" diperbarui untuk menunjuk ke buku berikutnya di *stack*.

```
public boolean spesial(){
        Buku current=top;
        while(current!=null){
            if(current.status=="Terlarang"){
                return true;
            }
            current=current.next;
        }
        return false;
    }
```

Script di atas adalah *method* "`special`" yang bertujuan untuk memeriksa apakah ada buku dalam *stack* yang memiliki status "`Terlarang`". *Method* ini menginisialisasi variabel "`current`" dengan nilai"`top`" yang menunjuk ke buku teratas dalam *stack*. Selama "`current`" tidak bernilai *null* berarti masih ada buku dalam *stack*. *Method* akan memeriksa apakah status buku saat ini sama dengan "`Terlarang`". Jika ditemukan buku dengan status "`Terlarang`", *method* akan mengembalikan nilai *true*. Jika tidak ada buku yang memenuhi

kriteria tersebut, maka setelah memeriksa semua buku, *method* akan mengembalikan nilai *false*.

```
public void tukar(String tittle1, String tittle2){
        if(tittle1==tittle2){
            System.out.println("Tidak perlu ditukar");
            return;
        }

        Buku prevP=null, currentP=top;
            while(currentP!=null){
                if(currentP.tittle==tittle1){
                    break;
                }
                prevP=currentP;
                currentP=currentP.next;
        }

        Buku prevQ=null, currentQ=top;
            while(currentQ!=null){
                if(currentQ.tittle==tittle2){
                    break;
                }
                prevQ=currentQ;
                currentQ=currentQ.next;
        }

        if(prevP!=null){
            prevP.next=currentQ;
        }else{
            top=currentQ;
        }

        if(prevQ!=null){
            prevQ.next=currentP;
        }else{
            top=currentP;
        }

        Buku bantu=currentP.next;
        currentP.next=currentQ.next;
        currentQ.next=bantu;
    }
```

Script di atas adalah *method* "`tukar`" yang berfungsi untuk menukar posisi dua buku dalam *stack* berdasarkan judulnya (*tittle1* dan *tittle2*). Pertama, *method* memeriksa apakah kedua judul yang diberikan sama. Jika iya, *method* akan mencetak pesan "`Tidak perlu ditukar`" dan keluar. *Method* ini menggunakan dua *loop while* untuk mencari buku yang sesuai dengan *tittle1* dan *tittle2*. Dalam *loop* pertama, variabel "`currentP`" menelusuri *stack* untuk menemukan buku dengan *tittle1*, sementara "`prevP`" menyimpan referensi ke buku sebelumnya. *Loop* kedua melakukan hal yang sama untuk *tittle2* dengan variable "`currentQ`" dan "`prevQ`". Setelah kedua buku ditemukan, *method* memeriksa apakah buku pertama ("`currentP`") dan buku kedua ("`currentQ`") ada dalam *stack*. Jika buku pertama

ada, maka "`prevP.next`" diatur untuk menunjuk ke buku kedua ("`currentQ`"). Jika tidak, "`top`" diatur untuk menunjuk ke buku kedua. Jika buku kedua ada, maka "`prevQ.next`" diatur untuk menunjuk ke buku pertama ("`currentP`"). Jika tidak, "`top`" diatur untuk menunjuk ke buku pertama. Lalu *method* melakukan pertukaran referensi antara "`currentP`" dan "`currentQ`". Terakhir "`current.next`" diatur untuk menunjuk ke buku yang berada di belakang"`currentQ`" dan"`current.next`" diatur untuk menunjuk ke buku yang berada di belakang"`currentP`".

```
public void tampilan(){
        Buku current=top;
        while(current!=null){
             System.out.println("=====================================
==");
             System.out.println("Judul Buku : " + current.tittle);
             System.out.println("Pengarang : " + current.author);
             System.out.println("Genre : " + current.genre);
             System.out.println("Status Buku : " + current.status);
             System.out.println("=====================================
==");
             current=current.next;
        }}}
```

Script di atas adalah *method* "`tampilan`" yang digunakan untuk menampilkan informasi tentang semua buku dalam *stack*. *Method* ini menginisialisasi variabel "`current`" dengan nilai "`top`" yang menunjuk ke buku teratas dalam *stack*. Kemudian, selama "`current`" tidak bernilai *null method* akan melakukan iterasi melalui setiap buku dalam *stack*. Dalam *loop while*, *method* mencetak "==========" dan menampilkan judul, pengarang, genre, dan status buku dengan mencetak masing-masing atribut menggunakan "`System.out.println()`".Lalu "`current`" diperbarui untuk menunjuk ke buku berikutnya dalam *stack*.

```
public class Queue {
   public Peminjam first;
   public Peminjam last;

   public void enQueue(String nama, String kartu){
        Peminjam dataBaru = new Peminjam(nama, kartu);
        if(first==null){
             first=dataBaru;
             last=dataBaru;
        }
        else{
             last.next=dataBaru;
             last=dataBaru;
        }}

   public void deQueue(){
        if(first==null){
             System.out.println("Data Kosong");
             return;
```

```
        }
        else{
            if(first.kartu=="Tidak Ada" && first.stack.spesial()==true){
                System.out.println("Tidak dapat meminjam buku");
            }
            else{
                first=first.next;
            }
      }
}
```

Script di atas adalah kelas "`Queue`" yang memiliki dua atribut yaitu "`first`" dan"`last`" yang menunjuk ke peminjam pertama dan terakhir dalam antrian. *Method* "`enQueue`" menambahkan peminjam baru ke belakang antrian. Jika antrian kosong, peminjam baru menjadi peminjam pertama dan terakhir. *Method* "`deQueue`" menghapus peminjam dari depan antrian dengan memeriksa apakah antrian kosong. Jika peminjam pertama memiliki kartu "`Tidak Ada`" dan memiliki buku dengan status "`Terlarang`", maka peminjam tidak dapat menghapus diri dari antrian. Jika tidak, peminjam pertama dihapus dengan memperbarui "`first`" untuk menunjuk ke peminjam berikutnya.

```
public void addBook(String tittle, String author, String genre, String
status){
        last.stack.push(tittle, author, genre, status);
        last.jumlah++;
    }

    public void hapus(String nama){
        Peminjam current=first;
        while(current!=null){
            if(current.next.nama==nama){
                current.next=current.next.next;
                return;
            }
            current=current.next;
        }
}
```

Script adalah *method* "`addBook`" digunakan untuk menambahkan buku baru ke dalam *stack* peminjam terakhir (*last*), yang menerima empat parameter yaitu *tittle*, *author*, genre, dan status. Lalu memanggil *method* "`push`" pada *stack* untuk menambahkan buku tersebut. Setelah menambahkan buku, *method* ini meanmbah atribut "`jumlah`" pada peminjam terakhir yang menghitung "`jumlah`" buku yang telah dipinjam. Kemudian, *method* "`hapus`" berfungsi untuk menghapus peminjam berdasarkan nama yang diberikan. *Method* ini menggunakan variabel "`current`" untuk menelusuri antrian peminjam mulai dari *first*. Jika ditemukan peminjam yang nama "`next`"-nya sama dengan nama yang dicari, maka peminjam tersebut dihapus dengan mengatur "`current.next`" menunjuk ke peminjam

berikutnya. Jika tidak ada peminjam yang ditemukan, *method* terus melanjutkan pencarian hingga akhir antrian.

```
public void tukar(String nama1, String nama2){
        if(nama1==nama2){
            System.out.println("Tidak perlu ditukar");
            return;
        }

        Peminjam prevP=null, currentP=first;
        while(currentP!=null){
            if(currentP.nama==nama1){
                break;
            }
            prevP=currentP;
            currentP=currentP.next;
        }

        Peminjam prevQ=null, currentQ=first;
        while(currentQ!=null){
            if(currentQ.nama==nama2){
                break;
            }
            prevQ=currentQ;
            currentQ=currentQ.next;
        }

        if(prevP!=null){
            prevP.next=currentQ;
        }else{
            first=currentQ;
        }

        if(prevQ!=null){
            prevQ.next=currentP;
        }else{
            first=currentP;
        }

        Peminjam bantu=currentP.next;
        currentP.next=currentQ.next;
        currentQ.next=bantu;
    }
```

Script method "`tukar`" digunakan untuk menukar posisi dua peminjam dalam antrian berdasarkan nama mereka (nama1 dan nama2). Pertama, *method* memeriksa apakah kedua nama sama. Jika iya maka menampilkan "`Tidak perlu ditukar`". Kemudian, dua *loop while* digunakan untuk mencari peminjam dengan nama yang sesuai dan menyimpan referensi ke peminjam sebelumnya. Setelah menemukan kedua peminjam, *method* memperbarui referensi "`next`" untuk menukar posisi mereka. Jika salah satu peminjam adalah peminjam pertama, maka "`first`" diatur sesuai kebutuhan sehingga referensi antara peminjam ditukar menggunakan variabel "`bantu`".

```
public void tampilanBuku(){
```

```
        System.out.println("=========================================");
          System.out.println("=\t\tBUKU  "  +  first.nama.toUpperCase()  +
"\t\t=");
         first.stack.tampilan();
    }
```

Script method "`tampilanBuku`" digunakan untuk menampilkan informasi buku yang dipinjam oleh peminjam pertama dalam antrian. *Method* ini pertama-tama mencetak garis pemisah. Kemudian, mencetak judul "`BUKU`" diikuti dengan nama peminjam pertama ("`first.nama`") yang diubah menjadi huruf kapital menggunakan "`toUpperCase()`". Setelah itu, *method* memanggil *method* "`tampilan()`" dari *stack* milik peminjam pertama ("`first.stack.tampilan()`") untuk menampilkan daftar buku yang telah dipinjam.

```
public void tampilanAntrian(){
        Peminjam current=first;
        int counter=0;
       System.out.println("=========================================");
        System.out.println("=             DAFTAR ANTRIAN            =");
        if(current==null){
            System.out.println("Antrian Kosong");
        }
        while(current!=null){
            current.antrian=++counter;
             System.out.println("=====================================
==");
            System.out.println("Nama : "+ current.nama);
            System.out.println("Antrian ke : "+ current.antrian);
            System.out.println("Jumlah Buku : "+ current.jumlah);
            System.out.println("Kartu Special : "+ current.kartu);
             System.out.println("=====================================
==");
            current=current.next;
        }}}
```

Script method "`tampilanAntrian`" berfungsi untuk menampilkan daftar peminjam dalam antrian beserta informasinya. *Method* ini memulai dengan menginisialisasi variabel "`current`" untuk menunjuk ke peminjam pertama dan "`counter`" untuk menghitung posisi peminjam dalam antrian. Setelah mencetak garis pemisah dan judul "`DAFTAR ANTRIAN`", *method* memeriksa apakah antrian kosong. Jika iya, pesan "`Antrian Kosong`" ditampilkan. Jika tidak kosong, *method* melakukan iterasi melalui setiap peminjam dalam antrian dengan menggunakan *loop while*, memperbarui atribut antrian setiap peminjam berdasarkan urutannya, dan mencetak nama, nomor antrian, jumlah buku yang dipinjam, serta kartu.

```
public class Main {
    public static void main(String[] args) {
        Queue queue = new Queue();

        queue.enQueue("Kazuma", "Tidak Ada");
       queue.addBook("Belajar Java", "Raysen", "Edukasi", "Buku Biasa");
         queue.addBook("Cara Menjadi Orang Kaya", "Teguh", "Fantasi",
"Buku Biasa");
```

```
        queue.enQueue("Hu Tao", "Ada");
         queue.addBook("Cara Tidur Cepat", "Teguh", "Edukasi Kayaknya",
"Terlarang");
        queue.addBook("Belajar C++", "Raysen", "Edukasi", "Buku Biasa");
           queue.addBook("Belajar Ilmu Hitam", "Mengumin", "Unknown",
"Terlarang");

        queue.enQueue("Kafka", "Tidak Ada");
        queue.addBook("Raysen the Forgotten One", "Unknown", "Sejarah",
"Terlarang");
         queue.addBook("Misteri Menghilangnya Nasi Puyung", "Optimus",
"Misteri", "Buku Biasa");
          queue.addBook("Cara Menjadi Milioner Dalam 1 Jam", "Master
Oogway", "Edukasi", "Buku Biasa");

        queue.enQueue("Xiangling", "Tidak Ada");
        queue.addBook("Unknown", "Unknown", "Unknown", "Buku Biasa");
```

Script di atas merupakan bagian dari kelas "`Main`", di mana sebuah objek "`Queue`" Bernama "`queue`" diinisialisasi untuk mengelola antrian peminjam dan buku. Di dalam *method* "`main`", beberapa peminjam seperti "`Kazuma`", "`Hu Tao`", "`Kafka`", dan "`Xiangling`" ditambahkan ke antrian menggunakan *method* "`enQueue`" dan setiap peminjam memiliki buku yang ditambahkan ke *stack* mereka melalui *method* "`addBook`" dengan berbagai atribut buku, termasuk status "`Buku Biasa`" dan "`Terlarang`".

```
queue.tampilanBuku();
        queue.deQueue();
        queue.tampilanAntrian();
        System.out.println("\n");

        queue.tampilanBuku();
        queue.deQueue();
        queue.tampilanAntrian();
        System.out.println("\n");

        queue.enQueue("Sucrose", "Ada");
        queue.addBook("Resurection", "Unknown", "Unknown", "Terlarang");
        queue.addBook("Alhcemy", "Albedo", "Sience", "Terlarang");
         queue.addBook("Durin The Forgotten Dragon", "Gold", "Misteri",
"Buku Biasa");

        queue.tampilanAntrian();
        System.out.println("\n");
```

Script di atas melanjutkan pengelolaan objek "`Queue`" di mana peminjam pertama dalam antrian ditampilkan bersama buku yang mereka pinjam menggunakan *method* "`tampilanBuku()`", kemudian peminjam tersebut dihapus dengan "`deQueue()`" dan status antrian ditampilkan kembali. Proses ini diulang untuk peminjam berikutnya. Setelah itu, peminjam baru Bernama "`Sucrose`" ditambahkan ke antrian dengan status "`Ada`" dan beberapa buku baru dengan status berbeda, termasuk "`Terlarang`" ditambahkan ke *stack* peminjam dan status antrian peminjam ditampilkan kembali.

```
queue.hapus("Xiangling");
        queue.tampilanAntrian();
        System.out.println("\n");

        queue.tukar("Sucrose", "Kafka");
        queue.tampilanAntrian();
        System.out.println("\n");

        queue.tampilanBuku();
        queue.deQueue();
        queue.tampilanAntrian();
        System.out.println("\n");

        queue.tampilanBuku();
        queue.deQueue();
        System.out.println("\n");

        queue.first.stack.pop();
        queue.tampilanBuku();
        System.out.println("\n");

        queue.deQueue();
        queue.tampilanAntrian();
        System.out.println("\n");
    }}
```

Script di atas melanjutkan operasi pada objek "`Queue`" dengan melakukan beberapa tindakan pengelolaan peminjam dan buku. Pertama, peminjam "`Xiangling`" dihapus dari antrian menggunakan *method* "`hapus`", diikuti dengan tampilan status antrian terkini. Selanjutnya, posisi peminjam "`Sucrose`" dan "`Kafka`" ditukar dan antrian ditampilkan setelah pertukaran. *Method* "`tampilanBuku()`" digunakan untuk menunjukkan buku yang dipinjam oleh peminjam pertama sebelum peminjam tersebut dihapus dengan "`deQueue()`". Proses ini diulang untuk peminjam berikutnya. Buku teratas di *stack* peminjam pertama dihapus menggunakan "`pop()`" dan status buku yang tersisa ditampilkan. Terakhir, peminjam terakhir dihapus dari antrian dan diakhiri dengan menampilkan status antrian yang telah diperbarui.