

LAPORAN TUGAS

MATA KULIAH PRAKTIKUM ALGOITMA DAN STRUKUR DATA

Dosen Pengampu : Vit Zuraida, S.Kom., M.Kom.

JOBSHEET QUEUE



Nama : Meisy Nadia Nababan

NIM : 2341760031

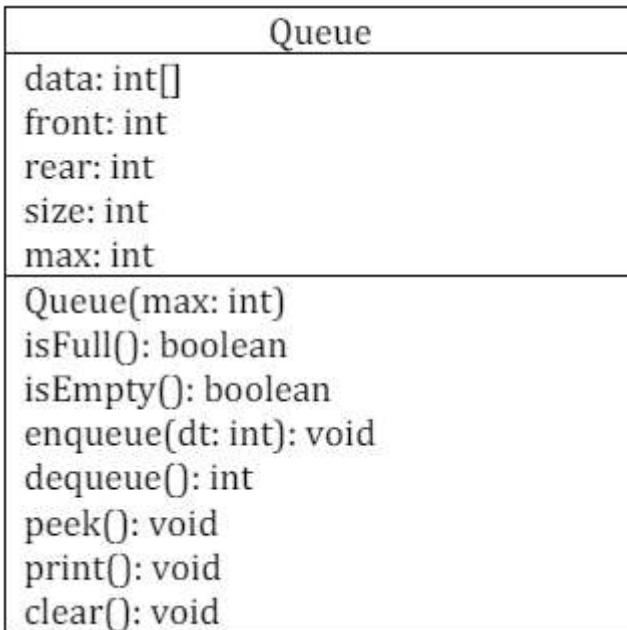
Prodi : D-IV Sistem Informasi Bisnis

**JURUSAN TEKNOLOGI
INFORMASI POLITEKNIK
NEGERI MALANG 2024**

8.2 Praktikum 1

8.2.1 Langkah-langkah Percobaan

1. Perhatikan Class Diagram untuk Queue berikut ini:



Berdasarkan diagram class tersebut, akan dibuat program class Queue dalam Java.

2. Buat folder dengan nama Praktikum08, kemudian buat class baru dengan nama Queue.

3. Tambahkan atribut-atribut Queue sesuai class diagram

```
public class Queue {  
    public int[] data;  
    public int max;  
    public int size;  
    public int front;  
    public int rear;  
}
```

4. Tambahkan constructornya

```
public Queue(int max) {  
    this.max = max;  
    this.data = new int[max];  
    this.size = 0;  
    this.front = this.rear = -1;  
}
```

5. Buat method isEmpty() yang digunakan untuk mengecek apakah queue kosong.

```

public boolean isEmpty() {
    return (size == 0);
}

```

6. Buat method isFull() yang digunakan untuk mengecek apakah queue sudah penuh.

```

public boolean isFull() {
    return (size == max);
}

```

7. Buat method peek() untuk menampilkan elemen queue pada posisi paling depan.

```

public void peek() {
    if (!isEmpty()) {
        System.out.println("Elemen terdepan: " + data[front]);
    } else {
        System.out.println("Queue kosong");
    }
}

```

8. Buat method print() untuk menampilkan seluruh elemen pada queue mulai dari posisi front sampai dengan posisi rear.

```

public void print() {
    if (isEmpty()) {
        System.out.println("Queue kosong");
    } else {
        int i = front;

        while (i != rear) {
            System.out.print(data[i] + " ");
            i = (i + 1) % max;
        }

        System.out.println(data[i] + " ");
        System.out.println("Jumlah elemen: " + size);
    }
}

```

9. Buat method clear() untuk menghapus semua elemen pada queue

```

public void clear() {
    front = rear = -1;
    size = 0;
}

```

10. Buat method enqueue() untuk menambahkan elemen dt ke dalam queue

```

public void enqueue(int dt) {
    if (isFull()) {
        System.out.println("Queue sudah penuh");
    } else {
        if (isEmpty()) {
            front = rear = 0;
        } else if (rear == max - 1) {
            rear = 0;
        } else {
            rear = rear + 1;
        }

        data[rear] = dt;
        size++;
    }
}

```

11. Buat method dequeue() untuk mengeluarkan data paling depan pada queue

```

public int dequeue() {
    int temp = 0;

    if (isEmpty()) {
        System.out.println("Queue masih kosong");
    } else {
        temp = data[front];
        size--;

        if (isEmpty()) {
            front = rear = -1;
        } else if (front == max - 1) {
            front = 0;
        } else {
            front++;
        }
    }

    return temp;
}

```

12. Buat file baru QueueDemo.java. Buat fungsi main

13. Deklarasikan Scanner dengan nama sc kemudian buat variabel kapasitas untuk menampung masukan berupa jumlah maksimal elemen yang dapat disimpan pada queue.

```

Scanner sc = new Scanner(System.in);
System.out.print("Masukkan kapasitas queue: ");
int kapasitas = sc.nextInt();

```

14. Lakukan instansiasi objek Queue dengan nama myQueue dengan argumen kapasitas

```

Queue myQueue = new Queue(kapasitas);

```

15. Lakukan perulangan menggunakan do-while untuk menampilkan daftar menu.

```
do {
    System.out.println("\nMasukkan operasi yang diinginkan");
    System.out.println("1. Enqueue");
    System.out.println("2. Dequeue");
    System.out.println("3. Print");
    System.out.println("4. Peek");
    System.out.println("5. Clear");
    System.out.println("6. Exit");
    System.out.println("-----");

    menu = sc.nextInt();
} while (menu >= 1 && menu <= 5);
```

16. Di dalam perulangan tersebut, terdapat switch-case untuk menjalankan operasi queue sesuai dengan masukan pilihan.

```
switch (menu) {
    case 1:
        System.out.print("Masukkan data baru:");
        int newData = sc.nextInt();
        myQueue.enqueue(newData);
        break;
    case 2:
        int deletedData = myQueue.dequeue();

        if (deletedData != 0) {
            System.out.println("Data yang dikeluarkan: " + deletedData);
        }

        break;
    case 3:
        myQueue.print();
        break;
    case 4:
        myQueue.peek();
        break;
    case 5:
        myQueue.clear();
        break;
}
```

17. Compile dan run program, kemudian amati hasilnya.

8.2.2 Verifikasi Hasil Percobaan

Samakan hasil run program Anda dengan gambar berikut ini.

Masukkan kapasitas queue: 6

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

1

Masukkan data baru:15

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

1

Masukkan data baru:23

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

3

15 23

Jumlah elemen: 2

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

4

Elemen terdepan: 15

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

2

Data yang dikeluarkan: 15

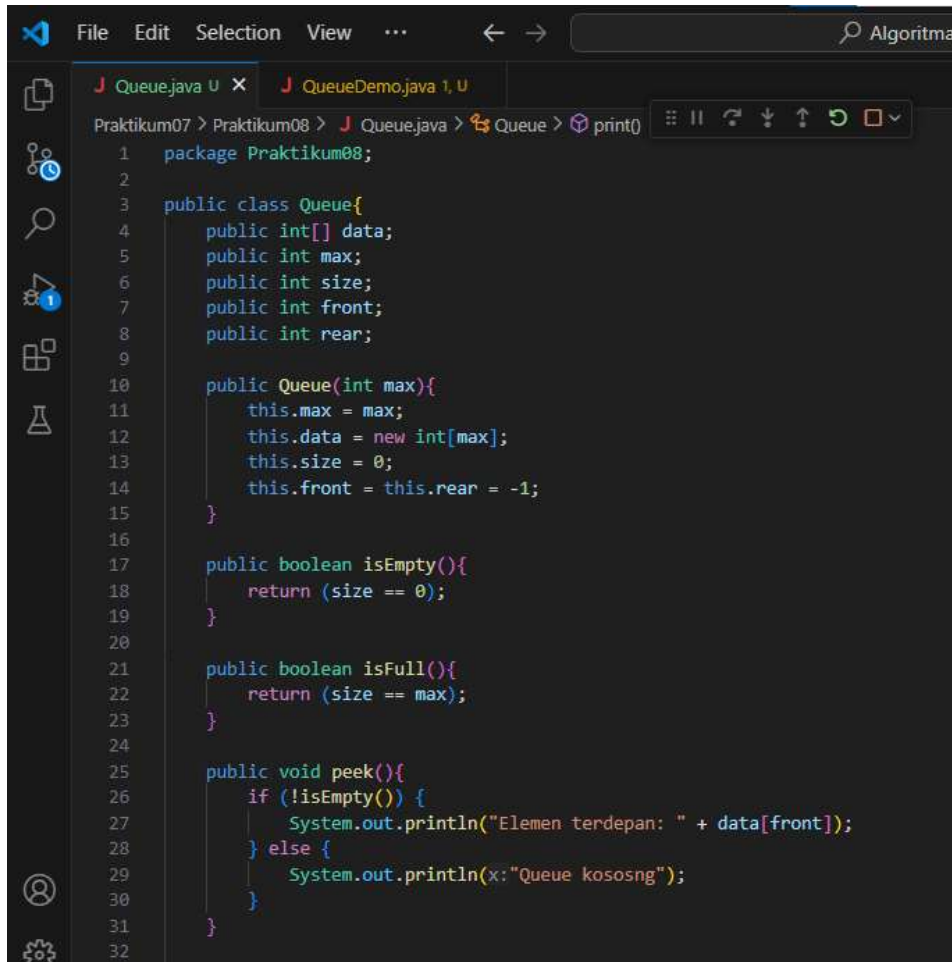
Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

3

23

Jumlah elemen: 1



The screenshot shows an IDE with two tabs: 'Queue.java' and 'QueueDemo.java'. The 'Queue.java' tab is active, displaying the following Java code:

```
1 package Praktikum08;
2
3 public class Queue{
4     public int[] data;
5     public int max;
6     public int size;
7     public int front;
8     public int rear;
9
10    public Queue(int max){
11        this.max = max;
12        this.data = new int[max];
13        this.size = 0;
14        this.front = this.rear = -1;
15    }
16
17    public boolean isEmpty(){
18        return (size == 0);
19    }
20
21    public boolean isFull(){
22        return (size == max);
23    }
24
25    public void peek(){
26        if (!isEmpty()) {
27            System.out.println("Elemen terdepan: " + data[front]);
28        } else {
29            System.out.println("Queue kosong");
30        }
31    }
32 }
```

```

33     public void print(){
34         if (isEmpty()) {
35             System.out.println(x:"Queue kosong");
36         } else {
37             int i = front;
38             while (i != rear) {
39                 System.out.print(data[i] + " ");
40                 i = (i + 1) % max;
41             }
42             System.out.println(data[i] + " ");
43             System.out.println("Jumlah elemen: " + size);
44         }
45     }
46
47     public void clear() {
48         front = rear = -1;
49         size = 0;
50     }

```

```

52     public void enqueue(int dt){
53         if (isFull()) {
54             System.out.println(x:"Queue sudah penuh");
55         } else {
56             if (isEmpty()) {
57                 front = rear = 0;
58             } else if(rear == max -1){
59                 rear = 0;
60             }else{
61                 rear = rear + 1;
62             }
63             data[rear] = dt;
64             size++;
65         }
66     }
67

```



```

68     public int dequeue(){
69         int temp = 0;
70
71         if (isEmpty()) {
72             System.out.println(x:"Queue masih kosong");
73         } else {
74             temp = data[front];
75             size--;
76
77             if (isEmpty()) {
78                 front = rear = -1;
79             } else if(front == max -1){
80                 front = 0;
81             }else{
82                 front++;
83             }
84         }
85         return temp;
86     }
87 }

```

J Queue.java U J QueueDemo.java 1, U X

Praktikum07 > Praktikum08 > J QueueDemo.java > QueueDemo

```

1  package Praktikum08;
2  import java.util.Scanner;
3
4  public class QueueDemo {
5      Run | Debug
6      public static void main(String[] args) {
7          Scanner sc = new Scanner(System.in);
8          System.out.print(s:"Masukkan kapasitas queue: ");
9          int kapasitas = sc.nextInt();
10
11          Queue myQueue = new Queue(kapasitas);
12          int menu;
13          do{
14              System.out.println(x:"\nMasukkan operasi yang diinginkan");
15              System.out.println(x:"1. Enqueue");
16              System.out.println(x:"2. Dequeue");
17              System.out.println(x:"3. Print");
18              System.out.println(x:"4. Peek");
19              System.out.println(x:"5. Clear");
20              System.out.println(x:"6. Exit");
21              System.out.println(x:"-----");

```

```

22     menu = sc.nextInt();
23     switch (menu) {
24         case 1:
25             System.out.print(s:"Masukkan data baru: ");
26             int newData = sc.nextInt();
27             myQueue.enqueue(newData);
28             break;
29         case 2:
30             int deletedData = myQueue.dequeue();
31
32             if (deletedData != 0) {
33                 System.out.println("Data yang dikeluarkan: " + deletedData);
34             }
35             break;
36         case 3:
37             myQueue.print();
38             break;
39         case 4:
40             myQueue.peek();
41             break;
42         case 5:
43             myQueue.clear();
44             break;
45     }
46 } while(menu >= 1 && menu <= 5);
47
48 }
49 }

```

Masukkan kapasitas queue: 6

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

1

Masukkan data baru: 15

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

1

Masukkan data baru: 23

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

3

15 23

Jumlah elemen: 2

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

4

Elemen terdepan: 15

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

2

Data yang dikeluarkan: 15

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

3

23

Jumlah elemen: 1

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit

|

8.2.3 Pertanyaan

1. Apakah nilai front selalu lebih kecil dibanding nilai rear? Jelaskan

Jawaban:

Tidak selalu. Nilai front tidak selalu lebih kecil dari nilai rear dalam struktur data queue yang diimplementasikan menggunakan array dengan pendekatan circular (lingkaran).

2. Pada method enqueue(), jelaskan maksud dan kegunaan dari potongan kode berikut

```
if (rear == max - 1) {  
    rear = 0;
```

Jawaban:

Potongan kode `if (rear == max - 1)` pada metode `enqueue()` digunakan untuk mengecek apakah rear sudah mencapai indeks maksimum dari array queue. Jika rear sudah berada di indeks maksimum, artinya rear telah mencapai akhir dari array dan tidak ada ruang kosong di belakang elemen terakhir. Dalam kasus ini, jika front masih berada di indeks 0 atau lebih tinggi (menunjukkan bahwa ada elemen yang telah dihapus di bagian depan queue), maka rear diatur kembali ke indeks awal array (`rear = 0`) untuk memanfaatkan ruang kosong di awal array.

3. Pada method dequeue(), jelaskan maksud dan kegunaan dari potongan kode berikut

```
if (front == max - 1) {  
    front = 0;
```

Jawaban:

Potongan kode `if (front == max - 1)` pada metode `dequeue()` digunakan untuk mengecek apakah front berada di indeks maksimum dari array queue. Jika front berada di indeks maksimum, artinya front sudah mencapai akhir array dan masih ada ruang kosong di awal array (karena beberapa elemen telah dihapus), maka front akan diatur kembali ke indeks awal array (`front = 0`) untuk memanfaatkan ruang kosong tersebut.

4. Pada method print, mengapa pada proses perulangan variabel i tidak dimulai dari 0 (`int i=0`), melainkan `int i=front`?

Jawaban:

Pada method print, perulangan dimulai dari nilai front karena front menunjukkan indeks elemen pertama dalam queue. Dengan memulai perulangan dari nilai front, kita dapat mencetak semua elemen queue dari elemen pertama hingga elemen terakhir, mengikuti urutan sebenarnya dalam queue.

5. Perhatikan kembali method print, jelaskan maksud dari potongan kode berikut

```
i = (i + 1) % max;
```

Jawaban:

Potongan kode `i = (i + 1) % max;` digunakan untuk menggerakkan indeks `i` ke elemen berikutnya dalam array queue. Ini dilakukan dengan menggunakan operasi modulus `%` terhadap `max`, yang akan mengembalikan sisa pembagian `i + 1` dengan `max`. Dengan demikian, `i` akan bergerak ke indeks

berikutnya dalam array, dan jika *i* mencapai max, ia akan kembali ke indeks 0 untuk melanjutkan perulangan di sepanjang array secara circular (lingkaran).

6. Apa yang dimaksud dengan queue overflow dan queue underflow? Tunjukkan potongan kode program di mana kondisi tersebut terjadi

Jawaban:

```
public static void main(String[] args) {  
    // Queue overflow  
    Queue queue = new Queue(max:3);  
    queue.enqueue(dt:1);  
    queue.enqueue(dt:2);  
    queue.enqueue(dt:3);  
    queue.enqueue(dt:4);  
  
    // Queue underflow  
    Queue myEmptyQueue = new Queue(max:3);  
    myEmptyQueue.dequeue();  
}
```

queue overflow adalah kondisi di mana antrian penuh dan tidak dapat lagi menampung elemen baru. Hal ini terjadi ketika Anda mencoba menambahkan elemen ke antrian yang sudah penuh.

queue underflow adalah kondisi di mana antrian kosong dan Anda mencoba menghapus elemen darinya. Hal ini terjadi ketika Anda mencoba mengeluarkan elemen dari antrian yang kosong.

8.3 Praktikum 2

8.3.1 Langkah-langkah Percobaan

1. Perhatikan Diagram Class berikut ini:

Kendaraan
noPlat: String noKartu: String saldo: double
Kendaraan(noPlat: String, noKartu: String, saldo: double) toString(): String

2. Buat folder dengan nama GerbangTol, kemudian buat class baru dengan nama Kendaraan

3. Tambahkan atribut-atribut Kendaraan berdasarkan class diagram di atas

```
public class Kendaraan {  
    public String noPlat;  
    public String noKartu;  
    public double saldo;  
}
```

4. Tambahkan pula konstruktornya seperti gambar berikut ini.

```
public Kendaraan(String noPlat, String noKartu, double saldo) {  
    this.noPlat = noPlat;  
    this.noKartu = noKartu;  
    this.saldo = saldo;  
}
```

5. Salin kode program class Queue pada Praktikum 1 untuk digunakan kembali pada Praktikum 2 ini.

Pada Praktikum 1, data yang disimpan dalam queue berupa array of integer, sedangkan pada Praktikum 2 data yang disimpan adalah array of object, maka perlu dilakukan modifikasi pada class Queue tersebut.

6. Lakukan modifikasi pada class Queue dengan mengubah tipe int[] data menjadi Kendaraan[] data karena pada kasus ini data yang akan disimpan pada queue berupa object-object bertipe Kendaraan.

```
public Kendaraan[] data;  
public int max;  
public int size;  
public int front;  
public int rear;  
  
public Queue(int max) {  
    this.max = max;  
    this.data = new Kendaraan[max];  
    this.size = 0;  
    this.front = this.rear = -1;  
}
```

7. Modifikasi method enqueue().

```

public void enqueue(Kendaraan dt) {
    if (isFull()) {
        System.out.println("Queue sudah penuh");
    } else {
        if (isEmpty()) {
            front = rear = 0;
        } else if (rear == max - 1) {
            rear = 0;
        } else {
            rear = rear + 1;
        }

        data[rear] = dt;
        size++;
    }
}

```

8. Modifikasi method dequeue().

```

public Kendaraan dequeue() {
    Kendaraan temp = null;

    if (isEmpty()) {
        System.out.println("Queue masih kosong");
    } else {
        temp = data[front];
        size--;

        if (isEmpty()) {
            front = rear = -1;
        } else if (front == max - 1) {
            front = 0;
        } else {
            front++;
        }
    }

    return temp;
}

```

9. Override method toString untuk class Kendaraan

```

public String toString() {
    return "No Plat: " + noPlat + ", No Kartu: " + noKartu + ", Saldo: " + saldo;
}

```

10. Selanjutnya, buat class baru dengan nama GerbangTolDemo.java pada folder GerbangTol. Buat fungsi main dengan while loop untuk menerima dan memilih menu oleh user


```

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    System.out.print("Masukkan kapasitas queue: ");
    int kapasitas = sc.nextInt();
    int menu;

    Queue myQueue = new Queue(kapasitas);

    do {
        System.out.println("\nMasukkan operasi yang diinginkan");
        System.out.println("1. Enqueue");
        System.out.println("2. Dequeue");
        System.out.println("3. Print");
        System.out.println("4. Peek");
        System.out.println("5. Clear");
        System.out.println("6. Exit");
        System.out.println("-----");

        menu = sc.nextInt();
        sc.nextLine();

    } while (menu >= 1 && menu <= 5);
}

```

11. Di dalam block DO, tambahkan switch case untuk menentukan fungsi queue yang dipanggil berdasarkan menu yang dipilih user

```

switch (menu) {
    case 1:
        System.out.println("Masukkan no plat: ");
        String noPlat = sc.nextLine();
        System.out.println("Masukkan no kartu: ");
        String noKartu = sc.nextLine();
        System.out.println("Masukkan saldo: ");
        double saldo = sc.nextDouble();

        Kendaraan newKendaraan = new Kendaraan(noPlat, noKartu, saldo);
        myQueue.enqueue(newKendaraan);
        break;

    case 2:
        Kendaraan deletedData = myQueue.dequeue();

        if (deletedData != null) {
            System.out.println("Data yang dikeluarkan: ");
            System.out.println(deletedData);
        }

        break;

    case 3:
        myQueue.print();
        break;

    case 4:
        myQueue.peek();
        break;

    case 5:
        myQueue.clear();
        break;
}

```

12. Compile dan jalankan class GerbangTolDemo, kemudian amati hasilnya.

8.3.2 Verifikasi Hasil Percobaan

Bandingkan hasil run program Anda dengan output berikut ini

```
Masukkan kapasitas queue: 2

Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
-----
1
Masukkan no plat:
A 1234 BC
Masukkan no kartu:
852304

Masukkan saldo:
20000

Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
-----
1
Masukkan no plat:
B 123 NM
Masukkan no kartu:
92934
Masukkan saldo:
40000

Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
-----
2
Data yang dikeluarkan:
No Plat: A 1234 BC, No Kartu: 852304, Saldo: 20000.0

Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
-----
4
Elemen terdepan: No Plat: B 123 NM, No Kartu: 92934, Saldo: 40000.0

Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
-----
3
No Plat: B 123 NM, No Kartu: 92934, Saldo: 40000.0
Jumlah elemen: 1
```

```
J Kendaraan.java U X J Queue.java U J GerbangTolDemo.java 1, U
Praktikum08 > GerbangTol > J Kendaraan.java > Kendaraan
1 //package GerbangTol;
2
3 public class Kendaraan {
4     public String noPlat;
5     public String noKartu;
6     public double saldo;
7
8     public Kendaraan(String noPlat, String noKartu, double saldo) {
9         this.noPlat = noPlat;
10        this.noKartu = noKartu;
11        this.saldo = saldo;
12    }
13
14    public String toString() {
15        return "No Plat : " + noPlat + ", No kartu : " + noKartu + ", Saldo : " + saldo;
16    }
17 }
```

```
J Kendaraan.java U X J Queue.java U X J GerbangTolDemo.java 1, U
Praktikum08 > GerbangTol > J Queue.java > Queue > enqueue
1 public class Queue{
2     public Kendaraan[] data;
3     public int max;
4     public int size;
5     public int front;
6     public int rear;
7
8     public Queue(int max){
9         this.max = max;
10        this.data = new Kendaraan[max];
11        this.size = 0;
12        this.front = this.rear = -1;
13    }
14
15    public boolean isEmpty(){
16        return (size == 0);
17    }
18
19    public boolean isFull(){
20        return (size == max);
21    }
22
23    public void peek(){
24        if (!isEmpty()) {
25            System.out.println("Elemen terdepan: " + data[front]);
26        } else {
27            System.out.println("Queue kosong");
28        }
29    }
}
```

```

29     }
30
31     public void print(){
32         if (isEmpty()) {
33             System.out.println(x:"Queue kosong");
34         } else {
35             int i = front;
36             while (i != rear) {
37                 System.out.print(data[i] + " ");
38                 i = (i + 1) % max;
39             }
40             System.out.println(data[i] + " ");
41             System.out.println("Jumlah elemen: " + size);
42         }
43     }
44
45     public void clear() {
46         front = rear = -1;
47         size = 0;
48     }

```

```

50     public void enqueue(Kendaraan dt){
51         if (isFull()) {
52             System.out.println(x:"Queue sudah penuh");
53         } else {
54             if (isEmpty()) {
55                 front = rear = 0;
56             } else if(rear == max -1){
57                 rear = 0;
58             }else{
59                 rear = rear + 1;
60             }
61             data[rear] = dt;
62             size++;
63         }
64     }

```

```

65
66     public Kendaraan dequeue() {
67         Kendaraan temp = null;
68
69         if (isEmpty()) {
70             System.out.println(x: "Queue masih kosong");
71         } else {
72             temp = data[front];
73             size--;
74
75             if (isEmpty()) {
76                 front = rear = -1;
77             } else if (front == max - 1) {
78                 front = 0;
79             } else {
80                 front++;
81             }
82         }
83         return temp;
84     }
85 }
86

```

J Kendaraan.java U J Queue.java U J GerbangTolDemo.java 1, U X

Praktikum08 > GerbangTol > J GerbangTolDemo.java > GerbangTol

```

1  //package GerbangTol;
2  import java.util.Scanner;
3
4  public class GerbangTolDemo {
5      public static void main(String[] args) {
6          Scanner sc = new Scanner(System.in);
7          System.out.print(s: "Masukkan kapasitas queue: ");
8          int kapasitas = sc.nextInt();
9          int menu;
10
11          Queue myQueue = new Queue(kapasitas);
12
13          do {
14              System.out.println(x: "\nMasukkan operasi yang diinginkan");
15              System.out.println(x: "1. Enqueue");
16              System.out.println(x: "2. Dequeue");
17              System.out.println(x: "3. Print");
18              System.out.println(x: "4. Peek");
19              System.out.println(x: "5. Clear");
20              System.out.println(x: "Exit");
21              System.out.println(x: "-----");
22
23              menu = sc.nextInt();
24              sc.nextLine();

```

```

26 switch (menu) {
27     case 1:
28         System.out.println(x: "Masukkan no plat:");
29         String noPlat = sc.nextLine();
30         System.out.println(x: "Masukkan no kartu: ");
31         String noKartu = sc.nextLine();
32         System.out.println(x: "Masukkan saldo: ");
33         double saldo = sc.nextDouble();
34
35         Kendaraan newKendaraan = new Kendaraan(noPlat, noKartu, saldo);
36         myQueue.enqueue(newKendaraan);
37         break;
38     case 2:
39         Kendaraan deletedData = myQueue.dequeue();
40         if (deletedData != null) {
41             System.out.println(x: "Data yang dikeluarkan: ");
42             System.out.println(deletedData);
43         }
44         break;
45     case 3:
46         myQueue.print();
47         break;
48     case 4:
49         myQueue.peek();
50         break;
51     case 5:
52         myQueue.clear();
53         break;
54 }
55
56 while (menu >= 1 && menu <= 5);

```

Masukkan kapasitas queue: 2

Masukkan operasi yang diinginkan

1. Enqueue
 2. Dequeue
 3. Print
 4. Peek
 5. Clear
- Exit

1

Masukkan no plat:

A 1234 BC

Masukkan no kartu:

852304

Masukkan saldo:

20000

Masukkan operasi yang diinginkan

1. Enqueue
 2. Dequeue
 3. Print
 4. Peek
 5. Clear
- Exit

1

Masukkan no plat:

B 123 NM

Masukkan no kartu:

92934

Masukkan saldo:

40000

Masukkan operasi yang diinginkan

1. Enqueue
 2. Dequeue
 3. Print
 4. Peek
 5. Clear
- Exit

2

Data yang dikeluarkan:

Data yang dikeluarkan:

No Plat : A 1234 BC, No kartu : 852304, Saldo : 20000.0

Masukkan operasi yang diinginkan

1. Enqueue
 2. Dequeue
 3. Print
 4. Peek
 5. Clear
- Exit

4

Elemen terdepan: No Plat : B 123 NM, No kartu : 92934, Saldo : 40000.0

```

Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
Exit
-----
3
No Plat : B 123 NM, No kartu : 92934, Saldo : 40000.0
Jumlah elemen: 1

```

8.3.3 Pertanyaan

1. Pada class Kendaraan, mengapa perlu dibuat method toString()? Cobalah comment fungsi tersebut kemudian jalankan program? Apakah terjadi compile error? Jika tidak, bagaimana output dari program?

```

14  /* @Override
15  public String toString() {
16      return "No Plat : " + noPlat + ", No kartu : " + noKartu + ", Saldo : " + saldo;
17  } */
18  }

```

Method toString() dalam class Kendaraan adalah method yang digunakan untuk mengembalikan representasi string dari objek Kendaraan. Representasi string ini adalah teks yang berisi informasi tentang objek Kendaraan, seperti nomor plat, nomor kartu, dan saldo.

Dengan mengomentari method toString(), program tidak akan menghasilkan kesalahan kompilasi karena method tersebut tidak digunakan dalam program. Namun, saat mencoba untuk mencetak objek Kendaraan, hasilnya akan menjadi representasi referensi objek GerbangTol.Kendaraan@. Hal ini karena method toString() tidak ada untuk memberikan representasi yang lebih bermakna dari objek Kendaraan.

```

Masukkan kapasitas queue : 1

Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
-----
1
Masukkan no plat : P 5678 BC
Masukkan no kartu : 2309
Masukkan saldo : 50000

```

```

Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
-----
2
Data yang dikeluarkan :
GerbangTol.Kendaraan@12edcd21

```



```

Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
-----
3
Queue kosong

Masukkan operasi yang diinginkan
1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Clear
6. Exit
-----
4
Queue kosong

```

2. Lakukan modifikasi program dengan menambahkan method baru bernama peekRear() pada class Queue yang digunakan untuk mengecek kendaraan yang berada di posisi paling belakang! Tambahkan pula menu baru Cek antrian paling belakang pada daftar menu dan lakukan pemanggilan method peekRear()

```

33     public void peekRear() {
34         if (!isEmpty()) {
35             System.out.println("Elemen paling belakang : " + data[rear]);
36         } else {
37             System.out.println(x:"Queue kosong");
38         }
39     }

12     do {
13         System.out.println(x:" \nMasukkan operasi yang diinginkan");
14         System.out.println(x:"1. Enqueue");
15         System.out.println(x:"2. Dequeue");
16         System.out.println(x:"3. Print");
17         System.out.println(x:"4. Peek");
18         System.out.println(x:"5. Peek rear");
19         System.out.println(x:"6. Clear");
20         System.out.println(x:"7. Exit");
21         System.out.println(x:"-----");

```

```

47         case 3:
48             myQueue.print();
49             break;
50         case 4:
51             myQueue.peek();
52             break;
53         case 5:
54             myQueue.peekRear();
55             break;
56         case 6:
57             myQueue.clear();
58     } while (menu >= 1 && menu <= 6);
59 }
60

```

Masukkan kapasitas queue : 2

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Peek rear
6. Clear
7. Exit

1

Masukkan no plat : A 345 DB

Masukkan no kartu : 948483

Masukkan saldo : 15000

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Peek rear
6. Clear
7. Exit

1

Masukkan no plat : C 875 JD

Masukkan no kartu : 937492

Masukkan saldo : 32000

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Peek rear
6. Clear
7. Exit

2

Data yang dikeluarkan :

No Plat : A 345 DB, No kartu : 948483, Saldo : 15000.0

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Peek rear
6. Clear
7. Exit

4

Elemen terdepan : No Plat : C 875 JD, No kartu : 937492, Saldo : 32000.0

Masukkan operasi yang diinginkan

1. Enqueue
2. Dequeue
3. Print
4. Peek
5. Peek rear
6. Clear
7. Exit

5

Elemen paling belakang : No Plat : C 875 JD, No kartu : 937492, Saldo : 32000.0

Tugas

Buatlah program antrian untuk mengilustrasikan antrian pasien di sebuah klinik. Ketika seorang pasien akan mengantri, maka dia harus mendaftarkan nama, nomor identitas, dan jenis kelamin.

Implementasikan class Pasien dan Queue sesuai class diagram berikut. Buatlah class KlinikDemo yang memuat daftar menu seperti pada praktikum di atas.

Pasien
nama: String noID: String jenisKelamin: char
Pasien (nama: String, noID: String, jenisKelamin: char) toString(): String

Implementasikan class Queue digambarkan sebagai berikut:

Queue
antrian: Pasien[] front: int rear: int size: int max: int
Queue(n: int) isEmpty(): boolean isFull(): boolean enqueue(dt: Pasien): void dequeue(): int print(): void peek(): void peekRear(): void peekPosition(nama: String): void daftarPasien(): void

Keterangan:

- Method peekRear(): digunakan untuk menampilkan data Pasien yang berada di posisi antrian paling belakang
- Method peekPosition(): digunakan untuk menampilkan seorang pasien (berdasarkan nama) posisi antrian ke berapa

```
Kendaraan.java U X Queue.java ...\GerbangTol U Pasien.java U X Queue.java ...\Tugas U KlinikDe
Praktiku C:\Users\HP\OneDrive\D ... KlinikDemo
1 Praktiku\Praktikum08\GerbangTol\Kendaraan.java • Untracked
2
3 public class Pasien {
4     String nama;
5     String noID;
6     char jenisKelamin;
7
8     public Pasien(String nama, String noID, char jenisKelamin){
9         this.nama = nama;
10        this.noID = noID;
11        this.jenisKelamin = jenisKelamin;
12    }
13
14    public String toString(){
15        return "Nama : " + nama + ", No.ID : " + noID + ",Jenis Kelamin : " + jenisKelamin;
16    }
17 }
```

```
3 public class Queue{
6     public int size;
7     public int front;
8     public int rear;
9
10    public Queue (int max) {
11        this.max = max;
12        this.antrian = new Pasien[max];
13        this.size = 0;
14        this.rear = -1;
15    }
16
17    public boolean isEmpty() {
18        return size == 0;
19    }
20
21    public boolean isFull() {
22        return size == max;
23    }
24
25    public void enqueue(Pasien dt) {
26        if (!isFull()) {
27            rear = (rear + 1) % max;
28            antrian[rear] = dt;
29            size++;
30            System.out.println("Pasien " + dt.nama + " telah ditambahkan ke dalam antrian");
31        } else {
32            System.out.println(x:"Antrian penuh, tidak dapat menambahkan pasien baru");
33        }
34    }
}
```

```

35
36  ✓ public Pasien dequeue() {
37  ✓     if (!isEmpty()) {
38         Pasien temp = antrian[front];
39         front = (front + 1) % max;
40         size--;
41         System.out.println("\nPasien " + temp.nama + "telah dipanggil dan dikeluarkan dari antrian");
42         return temp;
43  ✓     } else {
44         System.out.println(x:"Antrian kosong, tidak ada pasien yang dapat dipanggil");
45         return null;
46     }
47 }
48
49  ✓ public void print() {
50  ✓     if (!isEmpty()) {
51         System.out.println(x:"\nDaftar Pasien di Antrian : ");
52  ✓         for (int i = 0; i < size; i++) {
53             System.out.println(antrian[(front + i) % max]);
54         }
55  ✓     } else {
56         System.out.println(x:"Antrian kosong");
57     }
58 }

```

```

59
60 public void peek() {
61  ⚡     if (!isEmpty()) {
62         System.out.println("\nPasien yang sedang dipanggil : \n" + antrian[front]);
63     } else {
64         System.out.println(x:"Antrian kosong");
65     }
66 }
67
68 public void peekRear() {
69     if (!isEmpty()) {
70         System.out.println("\nPasien yang berada di posisi antrian paling belakang : \n" + antrian[rear]);
71     } else {
72         System.out.println(x:"Antrian kosong");
73     }
74 }

```

```

76     public void peekPosition(Pasien pasien) {
77         if (!isEmpty()) {
78             for (int i = 0; i < size; i++) {
79                 if (antrian[(front + i) % max].nama.equals(pasien)){
80                     System.out.println("\nPasien " + pasien.nama + "berada di posisi antrian ke-" + (i + 1));
81                     return;
82                 }
83             }
84             System.out.println("\nPasien dengan nama " +pasien.nama + "tidak ditemukan dalam antrian");
85         } else {
86             System.out.println(x:"Antrian kosong");
87         }
88     }
89
90     public void daftarPaien() {
91         if (!isEmpty()) {
92             System.out.println(x:"\nDaftar Pasien di Antrian");
93             for ( int i = 0; i < size; i++) {
94                 System.out.println((i + 1) + ". " + antrian[(front + i) % max].nama);
95             }
96         } else {
97             System.out.println(x:"Antrian kosong");
98         }
99     }
100
101     public void clear() {
102         front = 0;
103         rear = -1;
104         size = 0;
105         System.out.println(x:"Antrian telah dikosongkan");
106     }
107 }

```

```

1 package Praktikum08.Tugas;
2
3 import java.util.Scanner;
4
5 public class KlinikDemo {
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         System.out.print(s:"Masukkan jumlah pasien yang akan didaftarkan : ");
9         int kapasitas = sc.nextInt();
10        int menu;
11
12        Queue myQueue = new Queue(kapasitas);
13
14        do {
15            System.out.println(x:"\nMasukkan operasi yang diinginkan");
16            System.out.println(x:"1. Input Data Pasien");
17            System.out.println(x:"2. Pasien yang telah dipanggil dan keluar dari antrian");
18            System.out.println(x:"3. Daftar pasien di antrian");
19            System.out.println(x:"4. Pasien yang sedang dipanggil");
20            System.out.println(x:"5. Mengosongkan antrian pasien");
21            System.out.println(x:"6. Exit");
22            System.out.println(x:"-----");
23
24            menu = sc.nextInt();
25            sc.nextLine();
26

```

```

27         switch (menu) {
28             case 1:
29                 System.out.print(s:"Masukkan nama                : ");
30                 String nama = sc.nextLine();
31                 System.out.print(s:"Masukkan nomor identitas    : ");
32                 String noID = sc.nextLine();
33                 System.out.print(s:"Masukkan jenis kelamin (L/P) : ");
34                 char jenisKelamin = sc.next().charAt(index:0);
35                 sc.nextLine();
36
37                 Pasien newPasien = new Pasien(nama, noID, jenisKelamin);
38                 myQueue.enqueue(newPasien);
39                 break;
40             case 2:
41                 Pasien deletedData = myQueue.dequeue();
42                 if (deletedData != null) {
43                     System.out.println(x:"Data yang dikeluarkan : ");
44                     System.out.println(deletedData);
45                 }
46                 break;
47             case 3:
48                 myQueue.print();
49                 break;
50             case 4:
51                 myQueue.peek();
52                 break;
53             case 5:
54                 myQueue.clear();
55         }
56     } while (menu >= 1 && menu <= 5);

```

Masukkan jumlah pasien yang akan didaftarkan : 2

Masukkan operasi yang diinginkan

Masukkan operasi yang diinginkan

1. Input Data Pasien
2. Pasien yang telah dipanggil dan keluar dari antrian
3. Daftar pasien di antrian
4. Pasien yang sedang dipanggil
5. Mengosongkan antrian pasien
6. Exit

1

Masukkan nama : Difan

Masukkan nomor identitas : 2341760041

Masukkan jenis kelamin (L/P) : L

Pasien Difan telah ditambahkan ke dalam antrian


```
Masukkan operasi yang diinginkan
1. Input Data Pasien
2. Pasien yang telah dipanggil dan keluar dari antrian
3. Daftar pasien di antrian
4. Pasien yang sedang dipanggil
5. Mengosongkan antrian pasien
6. Exit
```

```
-----
3
```

Daftar Pasien di Antrian :

Nama : Meisy, No.ID : 2341760031,Jenis Kelamin : P

Nama : Difan, No.ID : 2341760041,Jenis Kelamin : L

```
Masukkan operasi yang diinginkan
1. Input Data Pasien
2. Pasien yang telah dipanggil dan keluar dari antrian
3. Daftar pasien di antrian
4. Pasien yang sedang dipanggil
5. Mengosongkan antrian pasien
6. Exit
```

```
-----
4
```

Pasien yang sedang dipanggil :

Nama : Meisy, No.ID : 2341760031,Jenis Kelamin : P

```
Masukkan operasi yang diinginkan
1. Input Data Pasien
2. Pasien yang telah dipanggil dan keluar dari antrian
3. Daftar pasien di antrian
4. Pasien yang sedang dipanggil
5. Mengosongkan antrian pasien
6. Exit
```

```
-----
2
```

Pasien Meisy telah dipanggil dan dikeluarkan dari antrian

Data yang dikeluarkan :

Nama : Meisy, No.ID : 2341760031,Jenis Kelamin : P

Masukkan operasi yang diinginkan

1. Input Data Pasien
2. Pasien yang telah dipanggil dan keluar dari antrian
3. Daftar pasien di antrian
4. Pasien yang sedang dipanggil
5. Mengosongkan antrian pasien
6. Exit

5

Antrian telah dikosongkan

Masukkan operasi yang diinginkan

1. Input Data Pasien
2. Pasien yang telah dipanggil dan keluar dari antrian
3. Daftar pasien di antrian
4. Pasien yang sedang dipanggil
5. Mengosongkan antrian pasien
6. Exit

6

PS C:\Users\HP\OneDrive\Dokumen\Algoritma dan Struktur Data Praktik> |