PRAKTIKUM 23
QUEUE

A. TUJUAN

Mahasiswa diharapkan mampu :

1. Memahami konsep Queue dan operasi-operasi pada queue
2. Memahami implementasi Queue pada Collection
3. Mengimplementasikan Queue menggunakan List
4. Mengimplementasikan Bounded Queue menggunakan Array

B. DASAR TEORI

Antrian (queue) adalah sekumpulan elemen, jika ada elemen baru yang ditambahkan, maka elemen tersebut akan berada di bagian belakang antrian Jika ada elemen yang harus dihapus atau keluar dari antrian, maka elemen yang keluar adalah elemen yang berada di sisi depan antrian. Atau konsep ini sering juga disebut dengan konsep FIFO (First In First Out).

Operasi yang terdapat pada queue adalah:

- Enqueue atau push adalah proses untuk memasukkan elemen artinya menambah data baru.
- Dequeue atau pop adalah proses untuk mengeluarkan elemen artinya menghapus data.
- Peek adalah proses untuk mengetahui elemen yang paling depan.

Array pada Queue

Array pada Queue adalah suatu array yang dibuat seakan-akan merupakan suatu garis lurus dengan satu pintu masuk dan satu pintu keluar. Pada gambar 1 merupakan implementasi queue menggunakan array. qfront untuk menandai elemen yang pertama, sedangkan qback untuk menambahkan elemen baru pada queue.

5	3	2		

qfront qback

5	3	2	10	

qfront qback Enqueue(10)

Dequeue() → elemen : 5

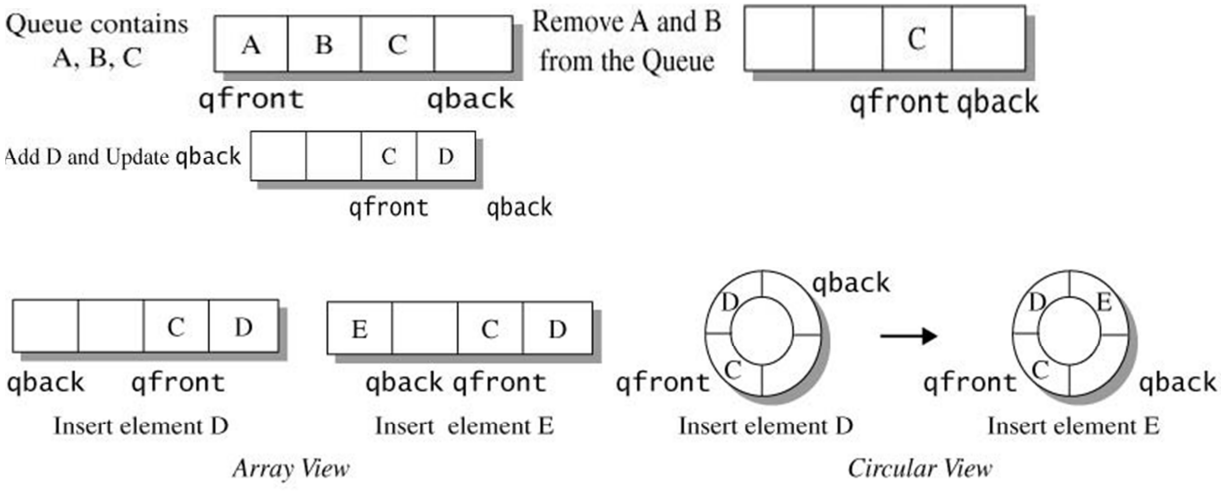
	3	2	10	
	qfront			qback

peek() → elemen : 3

Circular Bounded Queue

Cara mensimulasikan antrian secara circular dalam array linear menggunakan arithmetic modular. Arithmetic modular menggunakan ekspresi rumus (X % N) untuk menjaga besarnya nilai X pada range 0:N-1. Jika indeks telah sampai pada N dengan penambahan atau pengurangan tersebut, maka indeks akan diset pada angka 0.

Hal yang sama juga dilakukan pada Front jika dilakukan pengambilan item dari antrian. Setelah mengambil item dari antrian, kita melakukan increment terhadap Front untuk penunjukan pada posisi sesudahnya. Apabila indeks telah berada pada N, maka indeks diset juga pada angka 0.



Move qback forward: qback = (qback + 1) % qcapacity;

Move qfront forward: qfront = (qfront + 1) % qcapacity;

C. TUGAS PENDAHULUAN

Jelaskan mengenai **Queue. Buatlah queue** dengan array, panjang 5. Kondisi awal queue adalah sebagai berikut:

5	3	2		
qfront			qback	

Lakukan langkah-langkah berikut:

- Enqueue(7)
- Enqueue(8)
- Enqueue(9)
- Dequeue()
- Peek()
- Dequeue()
- Enqueue(9)

D. PERCOBAAN

D.1 Queue Collection

Percobaan 1 : LinkedList menerapkan interface Queue (1)

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

public class MainDemo {

  public void queueExample() {

    Queue queue = new LinkedList();

    queue.add("Java");
    queue.add("DotNet");
    queue.offer("PHP");
    queue.offer("HTML");

    System.out.println("remove: " + queue.remove());
    System.out.println("element: " + queue.element());
    System.out.println("poll: " + queue.poll());
    System.out.println("peek: " + queue.peek());
  }

  public static void main(String[] args) {
    new MainDemo().queueExample();
  }
```

```
}
```

Percobaan 2 : LinkedList menerapkan interface Queue (2)

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

public class Main {
  public static void main(String[] args) {
    Queue<String> myQueue = new LinkedList<String>();
    myQueue.add("A");
    myQueue.add("B");
    myQueue.add("C");
    myQueue.add("D");

    List<String> myList = new ArrayList<String>(myQueue);

    for (Object theFruit : myList)
      System.out.println(theFruit);
  }
}
```

D.2 Implementasikan Interface Queue menggunakan List

Buatlah interface Queue seperti dibawah ini:

Interface Queue	
boolean isEmpty()	Mengembalikan nilai true jika queue kosong dan false jika queue terdapat minimal satu elemen.
T peek()	Mengembalikan elemen di depan queue. Jika queue kosong melempar exception yaitu: throws NoSuchElementException.
T pop()	Menghapus elemen di depan queue dan mengembalikan nilainya. Jika queue kosong maka melempar exception yaitu : NoSuchElementException.
void push(T item)	Menyisipkan item di akhir queue
int size()	Mengembalikan jumlah elemen dari queue.

Isilah method-method yang merupakan implementasi dari interface Queue

```
import java.util.LinkedList;

public class LinkedQueue<T> implements Queue<T>{

  private LinkedList<T> qlist = null;
  public LinkedQueue ()
  {
    qlist = new LinkedList<T>();
```

```
    }

    @Override
    public boolean isEmpty() {
        throw new UnsupportedOperationException("Not supported yet.");
    }

    @Override
    public T peek() {
        return qlist.getFirst();
    }

    @Override
    public T pop() {
        throw new UnsupportedOperationException("Not supported yet.");
    }

    @Override
    public void push(T item) {
        throw new UnsupportedOperationException("Not supported yet.");
    }

    @Override
    public int size() {
        throw new UnsupportedOperationException("Not supported yet.");
    }
}
```

Selanjutnya, ujilah class LinkedQueue dengan program dibawah ini.

```
public class TestQueue {
    public static void main(String[] args) {
        LinkedQueue<String> queue = new LinkedQueue<String>();
        System.out.println(queue.isEmpty());

        queue.push("Java");
        queue.push("DotNet");
        queue.push("PHP");
        queue.push("HTML");

        System.out.println("remove: " + queue.pop());
        System.out.println("peek: " + queue.peek());
        System.out.println("size: " + queue.size());
    }
}
```

D.3 Implementasikan Interface Bounded Queue menggunakan Array dan Circular Bounded Queue

Bounded Queue adalah queue yang berisi elemen dengan size tertentu. Implementasikan interface BQueue yang merupakan perluasan dari Interface Queue

Interface BQueue extends Queue	
boolean full()	Mengembalikan nilai true jika queue penuh sesuai dengan size yang ditentukan dan false jika queue belum penuh.

```
public class ArrQueue<T> implements BQueue<T>{
   private T Arr[] ;
   private int qfront = 0 ;
   private int qback = 0 ;
   private int qcapacity = 0 ;

   public ArrQueue() {
      Arr = (T[]) new Object[50];
      qcapacity = 50;
   }


   public ArrQueue(int size) {
      Arr = (T[]) new Object[size];
      qcapacity = size;
   }

   @Override
   public boolean isEmpty() {
      throw new UnsupportedOperationException("Not supported yet.");
   }

   @Override
   public T peek() {
      throw new UnsupportedOperationException("Not supported yet.");
   }

   @Override
   public T pop() {
      throw new UnsupportedOperationException("Not supported yet.");
   }

   @Override
   public void push(T item) {
      throw new UnsupportedOperationException("Not supported yet.");
   }

   @Override
   public int size() {
      throw new UnsupportedOperationException("Not supported yet.");
   }

   @Override
```

```
    public boolean full() {
        throw new UnsupportedOperationException("Not supported yet.");
    }
}
```

Selanjutnya, ujilah class LinkedQueue dengan program dibawah ini.

```
public class Main {
    public static void main(String[] args) {
        try {
            ArrQueue<Integer> arr = new ArrQueue<Integer>(5);
            arr.push(1);
            arr.push(4);
            arr.push(10);
            arr.push(2);
            arr.push(7);
            arr.pop();
            arr.pop();
            arr.push(17);
            arr.push(3);
            arr.push(27);

        } catch (NoSuchElementException e) {
            System.out.println(e.getMessage());
        }
    }
}
```

E. LATIHAN

Ilustrasikan menggunakan gambar untuk bounded queue dibawah ini:

```
public class Main {
    public static void main(String[] args) {
        try {
            ArrQueue<Integer> arr = new ArrQueue<Integer>(5);
            arr.push(1);
            arr.push(4);
            arr.push(10);
            arr.push(2);
            arr.push(7);
            arr.pop();
            arr.pop();
            arr.push(17);
            arr.push(3);
            arr.push(27);

        } catch (NoSuchElementException e) {
            System.out.println(e.getMessage());
```

```
    }
  }
}
```

F. LAPORAN RESMI

Kerjakan hasil percobaan(D) dan latihan(E) di atas dan tambahkan analisa.