

# **LAPORAN PRAKTIKUM**

## **MODUL VII “QUEUE”**



**Disusun oleh:**  
**Andika Indra Prastawa**  
**NIM: 2311102033**

**Dosen Pengampu:**  
Wahyu Andi Saputra, S.Pd.,M.Eng.

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS INFORMATIKA  
INSTITUT TEKNOLOGI TELKOM PURWOKERTO  
PURWOKERTO  
2023**

# **BAB I**

## **TUJUAN PRAKTIKUM**

1. Mahasiswa mampu menjelaskan definisi dan konsep dari double queue
2. Mahasiswa mampu menerapkan operasi tambah, menghapus pada queue
3. Mahasiswa mampu menerapkan operasi tampil data pada queue

## **BAB II**

### **DASAR TEORI**

Antrian (Queue) merupakan kumpulan data yang mana penambahan elemen hanya bias dilakukan pada suatu ujung yaitu rear /tail / belakang, dan penghapusan dilakukan melalui ujung yang lainnya yaitu front / head / depan. Antrian disebut FIFO (First In First Out) yaitu elemen yang lebih dulu disisipkan merupakan elemen yang akan lebih dulu diambil.

Operasi-operasi dasar dari sebuah queue adalah :

1. Enqueue : proses penambahan elemen di posisi belakang
2. Dequeue : proses pengambilan elemen di posisi depan

Selain operasi dasar di atas, ada pula operasi-operasi lain yang dapat dilakukan terhadap sebuah queue yaitu :

1. Operasi pemeriksaan queue kosong (fungsi kosong)
2. Operasi pemeriksaan queue penuh (fungsi penuh).
3. Operasi inisialisasi queue (fungsi inisialisasi)

#### **Karakteristik Antrian**

Karakteristik antrian sebagai berikut :

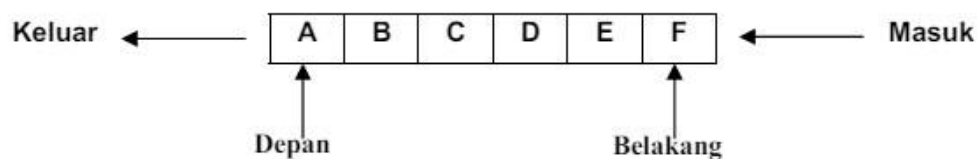
- Elemen antrian yaitu item-item data yang terdapat di elemen antrian
- Front (elemen terdepan dari antrian)
- Tail (elemen terakhir dari antrian)
- Count (jumlah elemen pada antrian)

- Status antrian apakah penuh atau kosong.

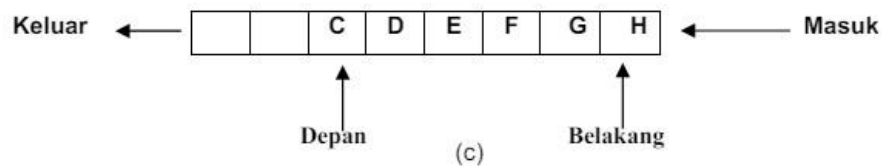
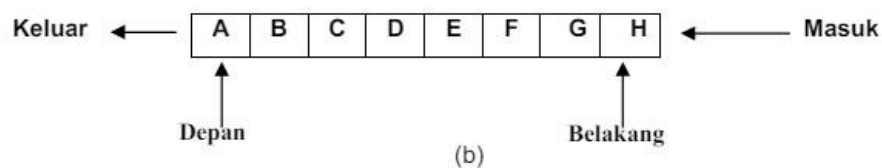
Penuh, jika elemen pada antrian mencapai kapasitas maximum antrian. Pada kondisi ini, tidak mungkin dilakukan penambahan ke antrian. Kosong, jika tidak ada elemen pada antrian. Pada kondisi ini, tidak mungkin dilakukan pengambilan elemen dari antrian.

## Implementasi Antrian

### Dengan menggunakan Array Statis



Jika ada elemen baru yang akan masuk pada gambar (diatas) maka ia akan diletakkan disebelah kanan F (gambar (b)). Jika ada elemen yang akan dihapus, maka A akan dihapus lebih dulu (gambar (c)).



## Penerapan Antrian

- Penjadwalan tugas dalam sistem operasi
- Transfer data dalam komunikasi jaringan

- Simulasi sistem dunia nyata (misalnya antrean tunggu)
- Antrian prioritas untuk pemrosesan acara Antrian untuk pemrosesan acara

## **Operasi pada Queue**

- enqueue() : menambahkan data ke dalam queue.
- dequeue() : mengeluarkan data dari queue.
- peek() : mengambil data dari queue tanpa menghapusnya.
- isEmpty() : mengecek apakah queue kosong atau tidak.
- isFull() : mengecek apakah queue penuh atau tidak.
- size() : menghitung jumlah elemen dalam queue.

## BAB III

### GUIDED

#### 1. Guided 1

##### Source code

```
#include <iostream>
using namespace std;

const int maksimalQueue = 5; // Maksimal antrian
int front = 0;                // Penanda antrian
int back = 0;                 // Penanda
string queueTeller[5];        // Fungsi pengecekan

bool isFull() {                // Pengecekan antrian penuh atau
    tidak
    if (back == maksimalQueue) {
        return true; // =1
    } else {
        return false;
    }
}

bool isEmpty() { // Antriannya kosong atau tidak
    if (back == 0) {
        return true;
    } else {
        return false;
    }
}

void enqueueAntrian(string data) { // Fungsi menambahkan antrian
    if (isFull()) {
        cout << "Antrian penuh" << endl;
    } else {
```

```

        if (isEmpty()) { // Kondisi ketika queue kosong
            queueTeller[0] = data;
            front++;
            back++;
        } else { // Antrianya ada isi
            queueTeller[back] = data;
            back++;
        }
    }
}

void dequeueAntrian() { // Fungsi mengurangi antrian
    if (isEmpty()) {
        cout << "Antrian kosong" << endl;
    } else {
        for (int i = 0; i < back; i++) {
            queueTeller[i] = queueTeller[i + 1];
        }
        back--;
    }
}

int countQueue() { // Fungsi menghitung banyak antrian
    return back;
}

void clearQueue() { // Fungsi menghapus semua antrian
    if (isEmpty()) {
        cout << "Antrian kosong" << endl;
    } else {
        for (int i = 0; i < back; i++) {
            queueTeller[i] = "";
        }
        back = 0;
    }
}

```

```

        front = 0;
    }
}

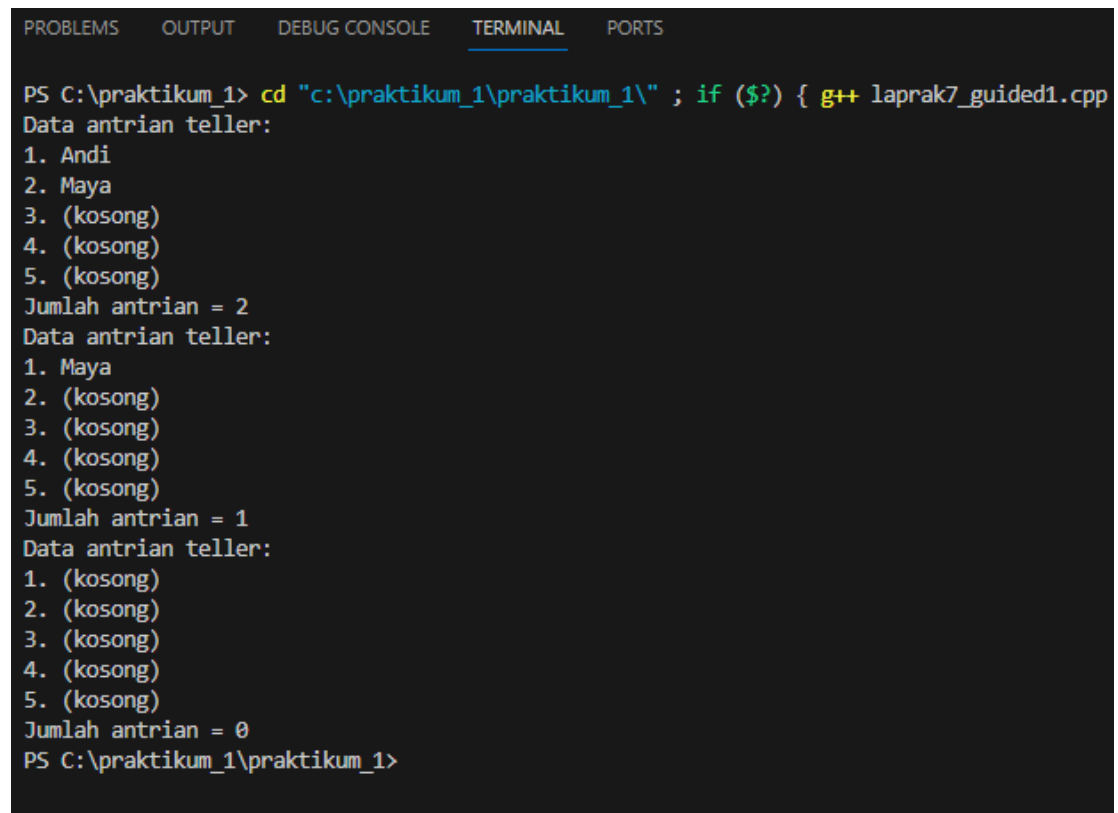
void viewQueue() { // Fungsi melihat antrian
    cout << "Data antrian teller:" << endl;
    for (int i = 0; i < maksimalQueue; i++) {
        if (queueTeller[i] != "") {
            cout << i + 1 << ". " << queueTeller[i] << endl;
        } else {
            cout << i + 1 << ". (kosong)" << endl;
        }
    }
}

int main() {
    enqueueAntrian("Andi");
    enqueueAntrian("Maya");
    viewQueue();
    cout << "Jumlah antrian = " << countQueue() << endl;
    dequeueAntrian();
    viewQueue();
    cout << "Jumlah antrian = " << countQueue() << endl;
    clearQueue();
    viewQueue();
    cout << "Jumlah antrian = " << countQueue() << endl;
    return 0;
}

```



## Screenshoot program



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS C:\praktikum_1> cd "c:\praktikum_1\praktikum_1\" ; if ($?) { g++ laprak7_guided1.cpp
Data antrian teller:
1. Andi
2. Maya
3. (kosong)
4. (kosong)
5. (kosong)
Jumlah antrian = 2
Data antrian teller:
1. Maya
2. (kosong)
3. (kosong)
4. (kosong)
5. (kosong)
Jumlah antrian = 1
Data antrian teller:
1. (kosong)
2. (kosong)
3. (kosong)
4. (kosong)
5. (kosong)
Jumlah antrian = 0
PS C:\praktikum_1\praktikum_1>
```

## Deskripsi program

Program ini merupakan implementasi dari antrian (queue) menggunakan array dalam bahasa pemrograman C++. Antrian ini dapat menyimpan data berupa nama seseorang. Program ini terdiri dari beberapa fungsi utama, yaitu `isFull()`, `isEmpty()`, `enqueueAntrian()`, `dequeueAntrian()`, `countQueue()`, `clearQueue()`, dan `viewQueue()`. Fungsi `isFull()` dan `isEmpty()` digunakan untuk memeriksa apakah antrian sudah penuh atau masih kosong. Fungsi `enqueueAntrian()` berfungsi untuk menambahkan data ke dalam antrian, sementara fungsi `dequeueAntrian()` digunakan untuk mengeluarkan data dari antrian. Fungsi `countQueue()` berguna untuk menghitung jumlah data yang ada di dalam antrian, dan fungsi `clearQueue()` digunakan untuk menghapus seluruh data dalam antrian. Fungsi `viewQueue()` digunakan untuk menampilkan data yang ada dalam antrian. Pada bagian `main()`, program menginisialisasi antrian, menambahkan beberapa data ke dalam antrian,

menampilkan data antrian, mengeluarkan data dari antrian, menghapus semua data dalam antrian, dan memeriksa jumlah data yang ada dalam antrian.

## LATIHAN KELAS - UNGUIDED

### 1. Unguided 1

#### Source code

```
#include <iostream>
using namespace std;
class Node
{
public:
    string data;
    Node *next;
    Node(string data)
    {
        this->data = data;
        this->next = nullptr;
    }
};
class ListQueue2311102033
{
private:
    Node *front;
    Node *back;

public:
    ListQueue2311102033()
    {
        front = nullptr;
        back = nullptr;
    }
    bool Kosong2311102033()
    {
        return front == nullptr;
    }
}
```

```

void enqueue(string data)
{
    Node *newNode = new Node(data);
    if (Kosong2311102033())
    {
        front = newNode;
        back = newNode;
    }
    else
    {
        back->next = newNode;
        back = newNode;
    }
}

void dequeue()
{
    if (Kosong2311102033())
    {
        cout << "Antrian Kosong" << endl;
        return;
    }
    Node *temp = front;
    front = front->next;
    if (front == nullptr)
    {
        back = nullptr;
    }
    delete temp;
}

int countQueue()
{
    int count = 0;
    Node *current = front;

```

```

        while (current != nullptr)
        {
            count++;
            current = current->next;
        }
        return count;
    }
    void clearQueue()
    {
        Node *current = front;
        while (current != nullptr)
        {
            Node *temp = current;
            current = current->next;
            delete temp;
        }
        front = nullptr;
        back = nullptr;
    }
    void viewQueue()
    {
        cout << "Data Antrian:" << endl;
        Node *current = front;
        int i = 1;
        while (current != nullptr)
        {
            cout << i << ". " << current->data << endl;
            current = current->next;
            i++;
        }
    }
};

```

```
int main()
{
    ListQueue2311102033 queueList;

    // menambahkan data antrian
    queueList.enqueue("Andika");
    queueList.enqueue("Indra");
    queueList.enqueue("Prastawa");

    // lihat antrian
    queueList.viewQueue();
    cout << "Jumlah antrian = " << queueList.countQueue() <<
endl;

    // mengeluarkan data antrian
    queueList.dequeue();
    // melihat update antrian
    cout << "\nUpdate Antrian:" << endl;
    queueList.viewQueue();
    cout << "Jumlah antrian = " << queueList.countQueue() <<
endl;

    // hapus antrian
    queueList.clearQueue();

    // periksa jika antrian kosong
    if (queueList.Kosong2311102033())
    {
        cout << "\nAntrian Kosong" << endl;
        cout << "Jumlah antrian = " << queueList.countQueue() <<
endl;
    }
    return 0;
}
```

## Screenshot program

```
PS C:\praktikum_1\praktikum_1> cd "c:\praktikum_1\praktikum_1\" ; if ($?) {  
tempCodeRunnerFile }  
Data Antrian:  
1. Andika  
2. Indra  
3. Prastawa  
Jumlah antrian = 3  
  
Update Antrian:  
Data Antrian:  
1. Indra  
2. Prastawa  
Jumlah antrian = 2  
  
Antrian Kosong  
Jumlah antrian = 0  
PS C:\praktikum_1\praktikum_1>
```

## Deskripsi program

Program ini mengimplementasikan antrian (queue) menggunakan linked list dalam bahasa pemrograman C++. Antrian ini terdiri dari node yang menyimpan data berupa string dan pointer ke node berikutnya. Kelas ListQueue2311102033 memiliki dua pointer utama yaitu front dan back untuk menandai awal dan akhir antrian. Fungsi utama dalam kelas ini meliputi: Kosong2311102033() untuk memeriksa apakah antrian kosong, enqueue() untuk menambahkan data ke antrian, dequeue() untuk mengeluarkan data dari antrian, countQueue() untuk menghitung jumlah data dalam antrian, clearQueue() untuk menghapus semua data dalam antrian, dan viewQueue() untuk menampilkan seluruh data dalam antrian. Pada bagian main(), program menambahkan beberapa data ke antrian, menampilkan isi antrian, mengeluarkan data dari antrian, menampilkan antrian yang telah diperbarui, menghapus semua data dalam antrian, dan memeriksa serta menampilkan status antrian apakah kosong atau tidak.

## 2. Unguided 1

### Source code

```
#include <iostream>
using namespace std;

class Node {
public:
    string name;
    string nim;
    Node *next;
    Node(string name, string nim) {
        this->name = name;
        this->nim = nim;
        this->next = nullptr;
    }
};

class QueueList {
private:
    Node *front;
    Node *back;

public:
    QueueList() {
        front = nullptr;
        back = nullptr;
    }

    bool isEmpty() {
        return front == nullptr;
    }

    void enqueue(string name, string nim) {
        Node* newNode = new Node(name, nim);
```



```

        if (isEmpty()) {
            front = newNode;
            back = newNode;
        } else {
            back->next = newNode;
            back = newNode;
        }
    }

void dequeue() {
    if (isEmpty()) {
        cout << "Antrian Kosong" << endl;
        return;
    }
    Node *temp = front;
    front = front->next;
    if (front == nullptr) {
        back = nullptr;
    }
    delete temp;
}

int countQueue() {
    int count = 0;
    Node *current = front;
    while (current != nullptr) {
        count++;
        current = current->next;
    }
    return count;
}

void clearQueue() {
    Node *current = front;

```

```

        while (current != nullptr) {
            Node *temp = current;
            current = current->next;
            delete temp;
        }
        front = nullptr;
        back = nullptr;
    }

    void viewQueue() {
        cout << "Data Antrian:" << endl;
        Node *current = front;
        int i = 1;
        while (current != nullptr) {
            cout << i << ". Name: " << current->name << ", NIM: " << current->nim << endl;
            current = current->next;
            i++;
        }
    }
};

int main() {
    QueueList queueList;

    // Tambahkan data antrian
    queueList.enqueue("Andika", "2311102033");
    queueList.enqueue("Indra", "2311102033");
    queueList.enqueue("Prastawa", "2311102033");

    // Tampilkan data antrian
    queueList.viewQueue();
    // menghitung jumlah antrian

```

```
        cout << "Jumlah antrian = " << queueList.countQueue() <<
endl;
        // hapus data antrian
        queueList.dequeue();

        // tampilkan data yang sudah di update
        cout << "\nUpdated Queue:" << endl;
        queueList.viewQueue();
        cout << "Jumlah antrian = " << queueList.countQueue() <<
endl;
        // hapus semua antrian
        queueList.clearQueue();

        // periksa antrian jika kosong
        if (queueList.isEmpty()) {
            cout << "\nAntrian Kosong" << endl;
            cout << "Jumlah antrian = " << queueList.countQueue() <<
endl;
        }
        return 0;
    }
}
```

## Screenshoot program

```
PS C:\praktikum_1\praktikum_1> cd "c:\praktikum_1\praktikum_1\" ; if ($?) { g++
prak7_unguided2 }
Data Antrian:
1. Name: Andika, NIM: 2311102033
2. Name: Indra, NIM: 2311102033
3. Name: Prastawa, NIM: 2311102033
Jumlah antrian = 3

Updated Queue:
Data Antrian:
1. Name: Indra, NIM: 2311102033
2. Name: Prastawa, NIM: 2311102033
Jumlah antrian = 2

Antrian Kosong
Jumlah antrian = 0
PS C:\praktikum_1\praktikum_1>
```

## Deskripsi program

Program ini merupakan implementasi struktur data antrian menggunakan linked list dalam bahasa pemrograman C++. Antrian ini dirancang untuk menyimpan data mahasiswa, yaitu nama dan NIM. Dua kelas utama didefinisikan dalam program ini, yaitu Node dan QueueList. Kelas Node mewakili elemen tunggal dalam linked list, sedangkan kelas QueueList mewakili antrian. Beberapa metode penting yang disediakan oleh QueueList antara lain: enqueue(string name, string nim) untuk menambahkan mahasiswa baru ke antrian, dequeue() untuk menghapus mahasiswa pertama dari antrian, countQueue() untuk menghitung jumlah mahasiswa dalam antrian, clearQueue() untuk menghapus semua mahasiswa dari antrian, dan viewQueue() untuk menampilkan data mahasiswa yang ada dalam antrian.

## **BAB IV**

### **KESIMPULAN**

Praktikum queue ini membahas konsep antrian (queue) dan implementasinya dalam bahasa pemrograman C++. Antrian merupakan struktur data yang menerapkan prinsip "first-in, first-out" (FIFO), yang berarti data yang masuk pertama akan keluar pertama. Dalam praktikum ini, kita mempelajari cara membuat antrian menggunakan array dan linked list. Pada implementasi dengan array, kita menetapkan ukuran array tertentu dan menggunakan variabel front dan back untuk menandai elemen pertama dan terakhir dalam antrian. Selain itu, kita juga membuat beberapa fungsi untuk mengelola antrian, seperti enqueue() untuk menambahkan data ke antrian, dequeue() untuk mengeluarkan data dari antrian, countQueue() untuk menghitung jumlah data dalam antrian, clearQueue() untuk menghapus semua data dalam antrian, dan viewQueue() untuk menampilkan data dalam antrian.

## **Daftar Pustaka**

- [1] Mardiani, G. T. (2014). *Queue*. Bandung: Unikom.
- [2] Karumanchi, N. (2016). Data Structures and algorithms made easy: Concepts, problems, Interview Questions. CareerMonk Publications.