

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL 7
“QUEUE”**



**DISUSUN OLEH :
RIZKY PERLINTA SEMBIRNG
2311102061**

**DOSEN
WAHYU ANDI SAPUTRA, S.PD., M.PD.**

**PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS INFORMATIKA
INSTITUT TEKNOLOGI TELKOM PURWOKERTO
2024**

A. DASAR TEORI

1. PENGENALAN QUEUE

i. Definisi dan Konsep Dasar Queue

Queue adalah struktur data yang mengikuti prinsip First-In-First-Out (FIFO), yang berarti elemen pertama yang masuk ke dalam antrian akan menjadi elemen pertama yang keluar dari antrian. Queue dapat diibaratkan sebagai antrian di mana elemen-elemen baru ditambahkan di satu ujung antrian (rear) dan elemen-elemen yang sudah ada dikeluarkan di ujung lainnya (front). Queue mirip dengan antrian nyata yang sering kita temui dalam kehidupan sehari-hari.

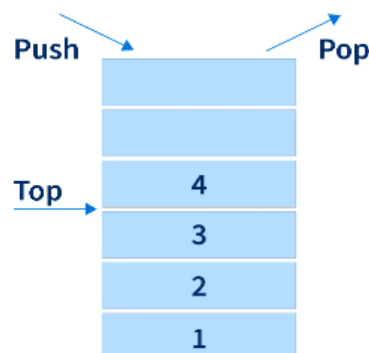
ii. Karakteristik Queue (FIFO — First-In-First-Out)

Karakteristik utama dari Queue adalah prinsip FIFO (First-In-First-Out). Elemen pertama yang dimasukkan ke dalam Queue akan menjadi elemen pertama yang diambil atau dihapus dari Queue. Elemen-elemen baru ditambahkan di ujung belakang Queue dan elemen-elemen yang sudah ada dikeluarkan dari ujung depan Queue. Dengan prinsip FIFO ini, Queue dapat membantu mengatur urutan data dan mempertahankan prioritas saat memproses elemen-elemen yang ada di dalamnya

iii. Perbedaan Stack dan Queue

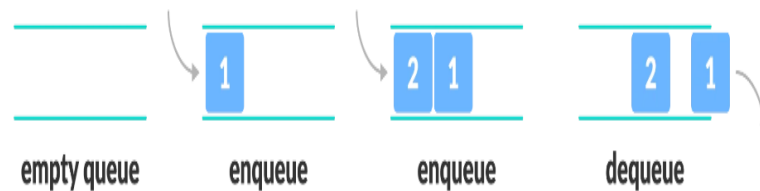
Perbedaan utama antara stack dan queue terletak pada cara penyisipan dan penghapusan elemen. Pada stack, kedua operasi dilakukan pada ujung yang sama, yaitu ujung atas. Sedangkan pada queue, kedua operasi dilakukan pada ujung yang berbeda, yaitu ujung depan dan ujung belakang.

- Ilustrasi dari Stack



Pada Stuck Ini mengikuti urutan LIFO (Last In First Out) untuk menyimpan elemen, artinya elemen yang dimasukkan terakhir akan keluar lebih dulu.

- **Ilustrasi dari Queue**



Berbeda dengan Stuck pada Queue Mengikuti urutan FIFO (First In First Out) untuk menyimpan elemen, artinya elemen yang dimasukkan terlebih dahulu akan keluar terlebih dahulu.

iv. Operasi Pada Queue

- `enqueue()` :menambahkandatakedalamqueue.
- `dequeue()` :mengeluarkandatadariqueue.
- `peek()` :mengambildatadariqueueetanpamenghapusnya.
- `isEmpty()` :mengecekapakahqueuekosongatautidak.
- `isFull()` :mengecekapakahqueuepenuhatautidak.
- `size()` :menghitungjumlahelemendalamqueue

B. Guided

Guided 1.

Source Code:

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

const int maksimalQueue = 5; // Batas maksimal antrian
int front = 0;                // Indeks awal antrian
int back = 0;                 // Indeks akhir antrian
string queueTeller[maksimalQueue]; // Array untuk menyimpan antrian

// Fungsi untuk memeriksa apakah antrian penuh
bool isFull()
{
    return back == maksimalQueue;
}

// Fungsi untuk memeriksa apakah antrian kosong
bool isEmpty()
{
    return back == 0;
}

// Fungsi untuk menambahkan elemen ke antrian
void enqueueAntrian(string data)
{
    if (isFull())
    {
        cout << "Antrian penuh" << endl;
    }
    else
    {
        queueTeller[back] = data;
        back++;
    }
}

// Fungsi untuk menghapus elemen dari antrian
void dequeueAntrian()
{
    if (isEmpty())
    {
        cout << "Antrian kosong" << endl;
    }
    else
    {
        for (int i = 0; i < back; i++)
```

```

        {
            queueTeller[i] = queueTeller[i + 1];
        }
        queueTeller[back - 1] = ""; // Membersihkan data terakhir
        back--;
    }
}

// Fungsi untuk menghitung jumlah elemen dalam antrian
int countQueue()
{
    return back;
}

// Fungsi untuk mengkosongkan semua elemen dalam antrian
void clearQueue()
{
    for (int i = 0; i < back; i++)
    {
        queueTeller[i] = "";
    }
    back = 0;
    front = 0;
}

// Fungsi untuk menampilkan semua elemen dalam antrian
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;
    }

```

Screenshot Output:

The screenshot shows the output of a C++ program and a text editor window. The program output displays a queue of names: 1. Andi, 2. Maya, 3. (kosong), 4. (kosong), 5. (kosong). The queue size is 2. Then, the queue is cleared, and the output shows: 1. Maya, 2. (kosong), 3. (kosong), 4. (kosong), 5. (kosong). The queue size is 1. Finally, the queue is cleared again, and the output shows: 1. (kosong), 2. (kosong), 3. (kosong), 4. (kosong), 5. (kosong). The queue size is 0.

The text editor window shows the following text:

```

Nama : Rizky Perlinta Sembiring
Nim : 2311102061
Kelas : IF-11

```

The text editor window also shows the file name "Na" and the menu options "File", "Edit", and "View". The status bar indicates "Ln 3, Col 14", "65 characters", "100%", "Window", and "UTF-8".

Deskripsi Program: program diatas adalah struktur data queue (antrian), Menggunakan sebuah array untuk menyimpan data antrian. Didalam kode diatas terdapat fungsi-fungsi seperti penambahan(enqueue antrian) dan penghapusan (dequeue antrian). Dalam output codingan diatas terlihat 2 data nama yaitu maya dan andi. Ketika kita menambahkan 3 data nama lagi maka nama andi dan maya akan tetap berada di posisi 1 dan 2 sebaliknya nama terakhir yg diinputkan akan akan terakhir keluar.

C. UNGUIDED

Unguided 1:

Source Code:

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

struct Node {
    string namaMahasiswa;
    string nim;
    Node* next;
};

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

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

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

    void enqueue(string namaMahasiswa, string nim) {
        Node* newNode = new Node{namaMahasiswa, nim, nullptr};
        if (isEmpty()) {
            front = newNode;
            back = newNode;
        } else {
            back->next = newNode;
            back = newNode;
        }
    }

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

```

        delete temp;
        cout<<"Antrian Berhasil Dihapus"<<endl;
    }
}

void viewQueue() {
    if (isEmpty()) {
        cout << "Antrian kosong." << endl;
    } else {
        cout << "Data antrian mahasiswa:" << endl;
        Node* current = front;
        int index = 1;
        while (current != nullptr) {
            cout << index << ". Nama: " << current->namaMahasiswa << ", NIM: " << current->nim << endl;
            current = current->next;
            index++;
        }
    }
}

void clearQueue() {
    while (!isEmpty()) {
        dequeue();
    }
}

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

};

void displayMenu() {
    cout << "\nMenu Antrian Teller:" << endl;
    cout << "1. Tambah Antrian" << endl;
    cout << "2. Hapus Antrian" << endl;
    cout << "3. Tampilkan Antrian" << endl;
    cout << "4. Bersihkan Antrian" << endl;
    cout << "5. Hitung Antrian" << endl;
    cout << "6. Keluar" << endl;
    cout << "Pilih operasi: ";
}

```



```

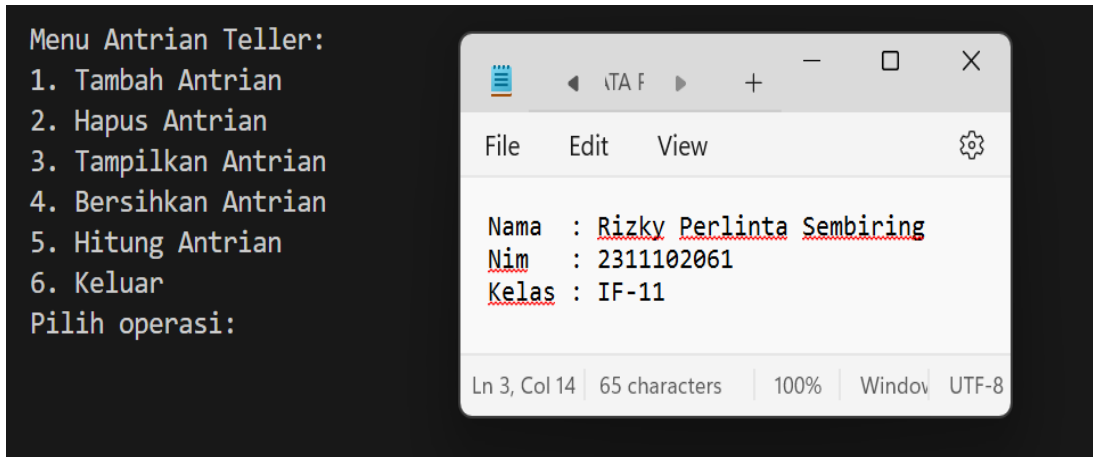
int main() {
    Queue q;
    int choice;
    string namaMahasiswa, nim;

    do {
        displayMenu();
        cin >> choice;
        switch (choice) {
            case 1:
                cout << "Masukkan Nama Mahasiswa: ";
                cin.ignore(); // Mengabaikan newline yang tersisa di
input buffer
                getline(cin, namaMahasiswa);
                cout << "Masukkan NIM: ";
                cin >> nim;
                q.enqueue(namaMahasiswa, nim);
                break;
            case 2:
                q.dequeue();
                break;
            case 3:
                q.viewQueue();
                break;
            case 4:
                q.clearQueue();
                break;
            case 5:
                cout << "Jumlah antrian = " << q.countQueue() <<
endl;
                break;
            case 6:
                cout << "Keluar dari program." << endl;
                break;
            default:
                cout << "Pilihan tidak valid. Silakan coba lagi." <<
endl;
        }
    } while (choice != 6);

    return 0;
}

```

Screenshot Output:



Deskripsi Program:

Program ini menggunakan linked list untuk mengimplementasikan struktur data queue antrian mahasiswa. Setiap mahasiswa direpresentasikan sebagai node dalam linked list, yang menyimpan informasi seperti nama dan NIM serta pointer ke node berikutnya. Operasi-operasi utama termasuk menambah, menghapus, menampilkan, membersihkan, dan menghitung jumlah mahasiswa dalam antrian.

Unguided 2:

Source Code:

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

struct Node {
    string namaMahasiswa;
    string nim;
    Node* next;
};

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

public:
    Queue() {
        front = nullptr;
        back = nullptr;
    }
};
```

```

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

void enqueue(string namaMahasiswa, string nim) {
    Node* newNode = new Node{namaMahasiswa, nim, nullptr};
    if (isEmpty()) {
        front = newNode;
        back = newNode;
    } else {
        back->next = newNode;
        back = newNode;
    }
}

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

void viewQueue() {
    if (isEmpty()) {
        cout << "Antrian kosong." << endl;
    } else {
        cout << "Data antrian mahasiswa:" << endl;
        Node* current = front;
        int index = 1;
        while (current != nullptr) {
            cout << index << ". Nama: " << current->namaMahasiswa << ", NIM: " << current->nim << endl;
            current = current->next;
            index++;
        }
    }
}

void clearQueue() {
    while (!isEmpty()) {
        dequeue();
    }
}

```

```

    }
}

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

};

void displayMenu() {
    cout << "\nMenu Antrian Mahasiswa:" << endl;
    cout << "1. Tambah Antrian" << endl;
    cout << "2. Hapus Antrian" << endl;
    cout << "3. Tampilkan Antrian" << endl;
    cout << "4. Bersihkan Antrian" << endl;
    cout << "5. Hitung Antrian" << endl;
    cout << "6. Keluar" << endl;
    cout << "Pilih operasi: ";
}

int main() {
    Queue q;
    int choice;
    string namaMahasiswa, nim;

    do {
        displayMenu();
        cin >> choice;
        switch (choice) {
            case 1:
                cout << "Masukkan Nama Mahasiswa: ";
                cin.ignore(); // Mengabaikan newline yang tersisa di
input buffer
                getline(cin, namaMahasiswa);
                cout << "Masukkan NIM: ";
                cin >> nim;
                q.enqueue(namaMahasiswa, nim);
                break;
            case 2:
                q.dequeue();
                break;
            case 3:
                q.viewQueue();
                break;

```

```

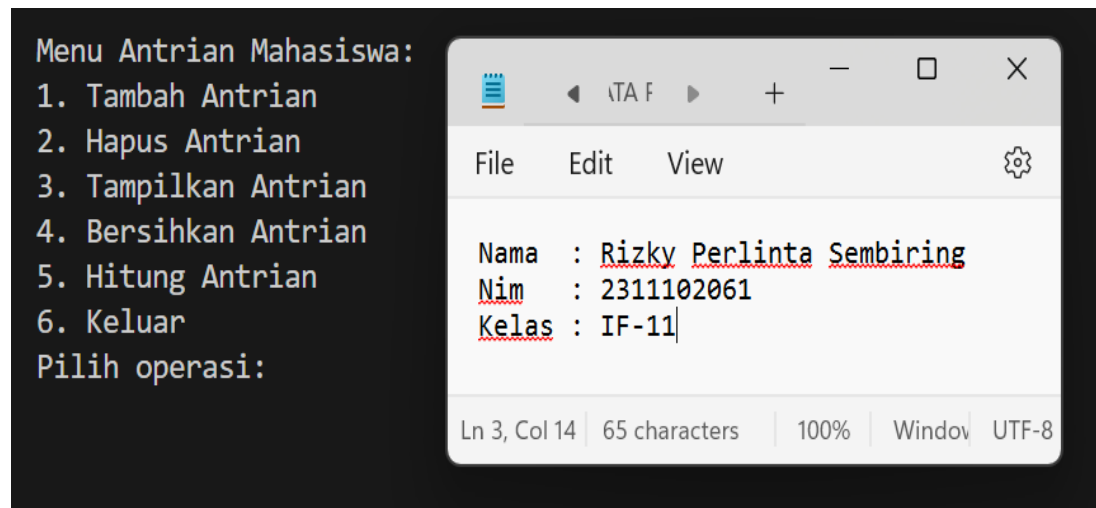
        case 4:
            q.clearQueue();
            break;
        case 5:
            cout << "Jumlah antrian = " << q.countQueue() <<
endl;
            break;
        case 6:
            cout << "Keluar dari program." << endl;
            break;
        default:
            cout << "Pilihan tidak valid. Silakan coba lagi." <<
endl;
    }
} while (choice != 6);

return 0;
}

```

Screenshot Output:

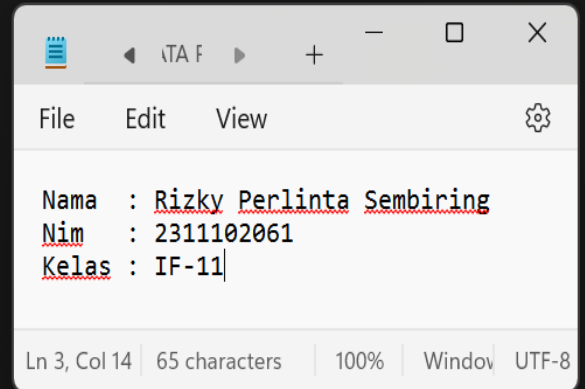
Menu:



Tambah Antrian:

Menu Antrian Mahasiswa:

1. Tambah Antrian
 2. Hapus Antrian
 3. Tampilkan Antrian
 4. Bersihkan Antrian
 5. Hitung Antrian
 6. Keluar
- Pilih operasi: 1
Masukkan Nama Mahasiswa: Rizky
Masukkan NIM: 2311102061



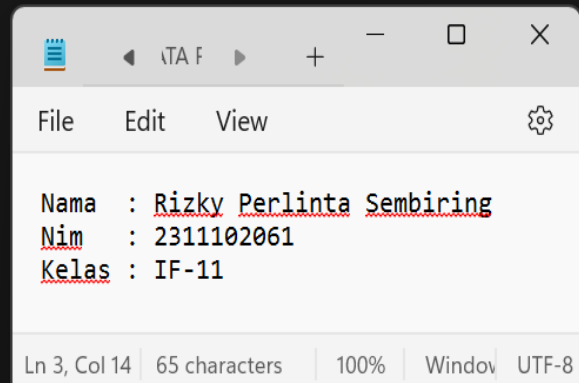
A screenshot of a text editor window with a menu bar (File, Edit, View) and a status bar (Ln 3, Col 14 | 65 characters | 100% | Window | UTF-8). The text content is:

```
Nama : Rizky Perlenta Sembiring
Nim  : 2311102061
Kelas : IF-11
```

Hitung antrian:

Menu Antrian Mahasiswa:

1. Tambah Antrian
 2. Hapus Antrian
 3. Tampilkan Antrian
 4. Bersihkan Antrian
 5. Hitung Antrian
 6. Keluar
- Pilih operasi: 5
Jumlah antrian = 3



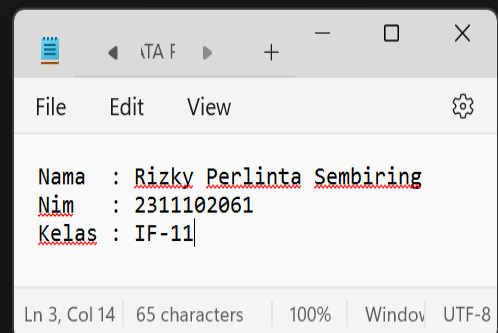
A screenshot of a text editor window with a menu bar (File, Edit, View) and a status bar (Ln 3, Col 14 | 65 characters | 100% | Window | UTF-8). The text content is:

```
Nama : Rizky Perlenta Sembiring
Nim  : 2311102061
Kelas : IF-11
```

Tampilkan antrian:

Menu Antrian Mahasiswa:

1. Tambah Antrian
 2. Hapus Antrian
 3. Tampilkan Antrian
 4. Bersihkan Antrian
 5. Hitung Antrian
 6. Keluar
- Pilih operasi: 3
Data antrian mahasiswa:
1. Nama: Rizky, NIM: 2311102061
2. Nama: Perlenta, NIM: 2311102062
3. Nama: Sembiring, NIM: 2311102063

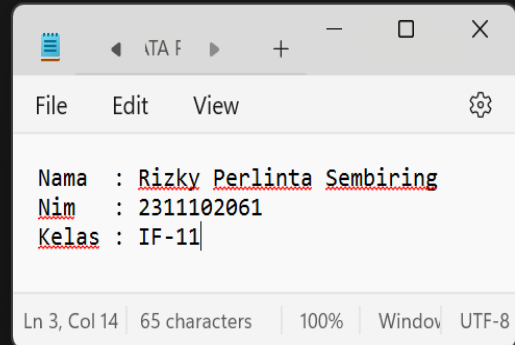


A screenshot of a text editor window with a menu bar (File, Edit, View) and a status bar (Ln 3, Col 14 | 65 characters | 100% | Window | UTF-8). The text content is:

```
Nama : Rizky Perlenta Sembiring
Nim  : 2311102061
Kelas : IF-11
```

Hapus antrian pertama:

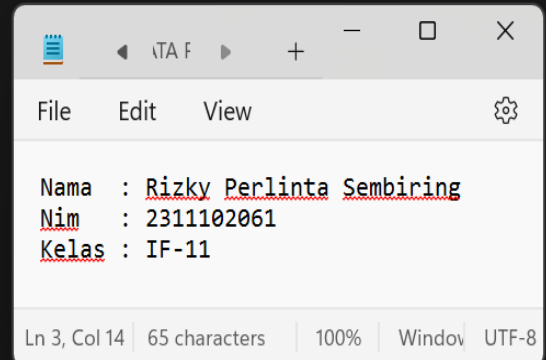
```
Menu Antrian Mahasiswa:
1. Tambah Antrian
2. Hapus Antrian
3. Tampilkan Antrian
4. Bersihkan Antrian
5. Hitung Antrian
6. Keluar
Pilih operasi: 2
Antrian Berhasil Dihapus
```



```
Menu Antrian Mahasiswa:
1. Tambah Antrian
2. Hapus Antrian
3. Tampilkan Antrian
4. Bersihkan Antrian
5. Hitung Antrian
6. Keluar
Pilih operasi: 3
Data antrian mahasiswa:
1. Nama: Perlinta, NIM: 2311102062
2. Nama: Sembiring , NIM: 2311102063
```

Bersihkan data antrian:

```
Menu Antrian Mahasiswa:
1. Tambah Antrian
2. Hapus Antrian
3. Tampilkan Antrian
4. Bersihkan Antrian
5. Hitung Antrian
6. Keluar
Pilih operasi: 4
Antrian Berhasil Dihapus
Antrian Berhasil Dihapus
```



```
Menu Antrian Mahasiswa:
1. Tambah Antrian
2. Hapus Antrian
3. Tampilkan Antrian
4. Bersihkan Antrian
5. Hitung Antrian
6. Keluar
Pilih operasi: 3
Antrian kosong.
```

Deskripsi Program:

Program ini merupakan implementasi sederhana dari struktur data queue sistem antrian untuk manajemen mahasiswa. Menggunakan struktur data queue, program memungkinkan pengguna untuk menambah antrian, menghapus antrian, menampilkan antrian, bersihkan antrian, hitung antrian, dan keluar. Ketika user menambahkan 3 data nama antrian contoh Rizky, Perlinta, Sembiring ketika user menghapus antrian maka nama yang terhapus adalah Rizky.

D. Kesimpulan

Perbedaan utama antara queue dan stack adalah bahwa stack mengikuti prinsip LIFO (Last In, First Out), di mana elemen terakhir yang dimasukkan adalah yang pertama dikeluarkan, sementara queue mengikuti prinsip FIFO (First In, First Out), di mana elemen pertama yang dimasukkan adalah yang pertama dikeluarkan. pada pengoperasian stack dan queue juga berbeda pada stack Operasi penyisipan dikenal sebagai push dan operasi penghapusan dikenal sebagai pop . sedangkan pada queue Operasi penyisipan disebut enqueue dan operasi penghapusan disebut dequeue .

E. Referensi

[1] Asisten Pratikum “Modul 7 Queue”, Learning Management System, 2024.

[2] RICZKY, PRATAMA, “Queue: Pengenalan, Implementasi, Operasi Dasar, dan Aplikasi”. (May 25, 2023), diakses pada 23 mei, dari

<https://medium.com/@furatamarizuki/queue-pengenalan-implementasi-operasi-dasar-dan-aplikasi-c5eed7e871a3>

[3] cahyasmara, “Antrian / Queue Pemograman C/C++”, (2017), diakses pada 23 mei, dari

<https://cahyasmara.blogspot.com/2017/06/antrian-queue-pemograman-cc-struktur.html>

[4] Chaudhary, Namita, “Difference Between Stack and Queue Data Structures”, (April 8, 2024), diakses pada 23 mei, dari

<https://www.scaler.com/topics/difference-between-stack-and-queue/>