LAPORAN PRAKTIKUM
MODUL 7 QUEUE

Disusun oleh:

Fahri Ramadhan

NIM : 2311102024

Dosen Pengampu:

Wahyu Andi Saputra, S.Pd., M.Eng.

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS INFORMATIKA

INSTITUT TEKNOLOGI TELKOM PURWOKERTO

2024

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

Pengertian Queue

Queue jika diartikan secara harfiah, queue berarti antrian, Queue merupakan suatu struktur data linear. Konsepnya hampir sama dengan Stack, perbedaannya adalah operasi penambahan dan penghapusan pada ujung yang bebeda. Pada Stack atau tumpukan menggunakan prinsip “Masuk terakhir keluar pertama”atau LIFO (Last In First Out), Maka pada Queue atau antrian prinsip yang digunakan adalah “Masuk Pertama Keluar Pertama” atau FIFO (First In First Out). Data-data di dalam antrian dapat bertipe integer, real, record dalam bentuk sederhana atau terstruktur.

Queue (antrian) adalah salah satu list linier dari struktur data yang beroperasi dengan cara First In First Out (FIFO) yaitu elemen pertama yang masuk merupakan elemen yang pertama keluar. Contohnya, ialah dalam sebuah antrian pembelian tiket bagi yang pertama masuk maka dia pulalah yang pertama keluar/selesai. Untuk penyisipan (INSERT) hanya dapat dilakukan pada satu sisi yaitu sisi belakang (REAR), sedangkan untuk penghapusan (REMOVE) pada sisi depan (FRONT) dari list.

Queue/antrian adalah ordered list dengan penyisipan di satu ujung, sedang penghapusan di ujung lain. Ujung penyisipan biasa disebut rear/tail, sedang ujung penghapusan disebut front/head. Fenomena yang muncul adalah elemen yang lebih dulu disisipkan akan juga lebih dulu diambil.

Queue merupakan kasus khusus ordered list. Dengan karakteristik terbatas itu maka kita dapat melakukan optimasi representasi ADT Queue untuk memperoleh kerja paling optimal..

GUIDED

BAB III

1. GUIDED 1 SOURCE CODE

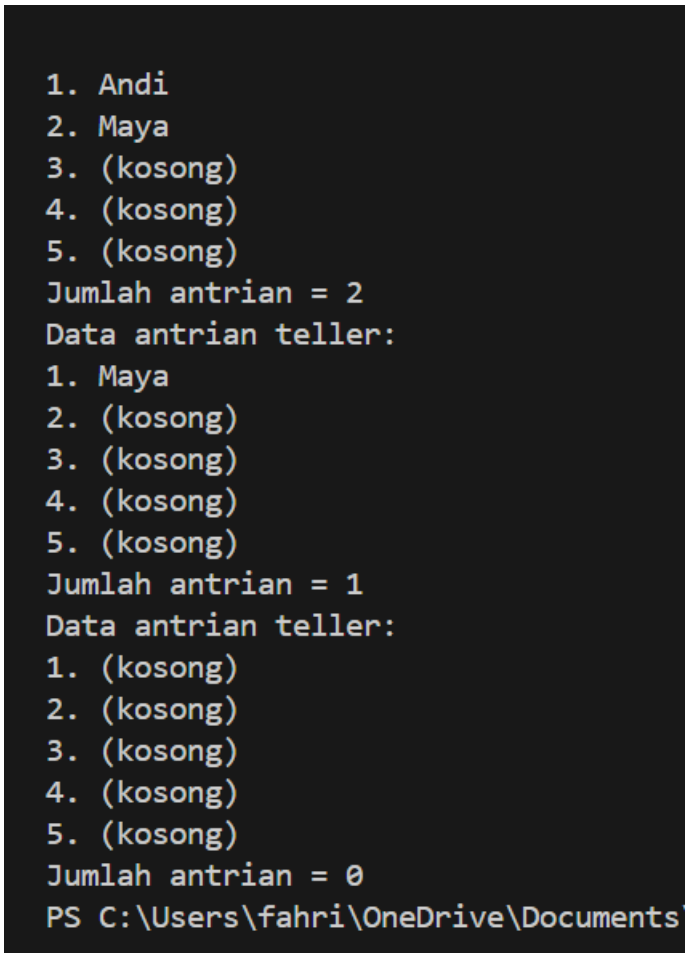
```
#include <iostream>
using namespace std;

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



DESKRIPSI PROGRAM

maksimalQueue adalah variabel konstan yang menentukan jumlah maksimum antrian yang dapat ditampung. front dan back adalah variabel yang digunakan untuk menandai posisi awal dan akhir antrian.

UNGUIDED

1. UNGUIDED 1

Ubahlah penerapan konsep queue pada bagian guided dari array menjadi linked list

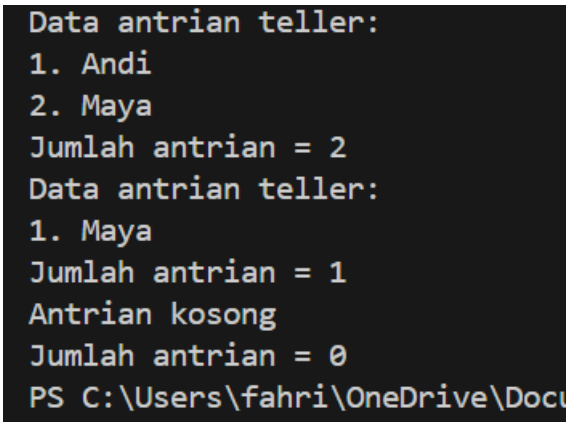
SOURCE CODE

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

struct Node {
string data;
    Node* next;
};
class Queue {
private:
    Node* front;
    Node* back;
 public:     Queue()
{          front =
NULL;          back =
NULL;
    }
    bool isEmpty() {
return front == NULL;
    }
    void enqueue(string data) {
Node* newNode = new Node;
newNode->data = data;
```

```
        newNode->next = NULL;


        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;
delete temp;
        }
    }
    int countQueue() {
int count = 0;
        Node* current = front;
while (current != NULL) {
count++;
            current = current->next;
        }
        return count;
    }
    void clearQueue() {
while (!isEmpty()) {
dequeue();
```

```
        }
    }
    void viewQueue() {
if (isEmpty()) {
            cout << "Antrian kosong" << endl;
        } else {
            cout << "Data antrian teller:" <<
endl;              Node* current = front;
int position = 1;             while (current !=
NULL) {
                cout << position << ". " << current->data <<
endl;
                current = current->next;
position++;
            }
        }
    }
};
int main() {     Queue
queue;
queue.enqueue("Andi");
queue.enqueue("Maya");
queue.viewQueue();
    cout << "Jumlah antrian = " << queue.countQueue() <<
endl;     queue.dequeue();     queue.viewQueue();
    cout << "Jumlah antrian = " << queue.countQueue() <<
endl;     queue.clearQueue();     queue.viewQueue();
    cout << "Jumlah antrian = " << queue.countQueue() << endl;
return 0;
}
```

SCREENSHOOT PROGRAM



DESKRIPSI PROGRAM

Fungsi enqueue() digunakan untuk menambahkan data ke antrian. Fungsi dequeue() digunakan untuk mengurangi data dari antrian. Fungsi countQueue() digunakan untuk menghitung jumlah data yang ada di antrian. Fungsi clearQueue() digunakan untuk menghapus semua data di antrian.

2. UNGUIDED 2

Buatlah program untuk menentukan apakah kalimat tersebut yang diinputkan dalam program stack adalah palindrom/tidak. Palindrom kalimat yang dibaca dari depan dan belakang sama. Jelaskan bagaimana cara kerja programnya

SOURCE CODE

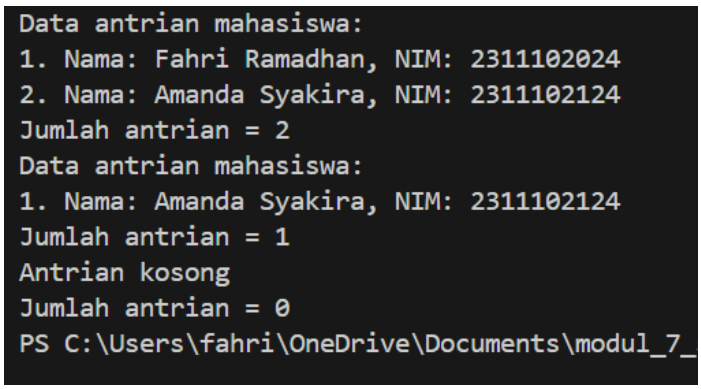
```
#include <iostream>
using namespace std;
  struct Node {
string nama;
```

```
    string nim;
Node* next;
};
class Queue {
private:
    Node* front;
Node* back;
 public:     Queue()
{          front =
NULL;          back =
NULL;
    }
    bool isEmpty() {
return front == NULL;
    }
    void enqueue(string nama, string nim)
{          Node* newNode = new Node;
newNode->nama = nama;          newNode->nim
= nim;          newNode->next = NULL;

        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;
delete temp; }
    }
    int countQueue() {
int count = 0;
        Node* current = front;
while (current != NULL) {
count++;
            current = current->next;
        }
        return count;
    }
    void clearQueue() {
while (!isEmpty()) {
dequeue();
        }
    }
    void viewQueue() {
if (isEmpty()) {
            cout << "Antrian kosong" << endl;
        } else {
            cout << "Data antrian mahasiswa:" <<
endl;             Node* current = front;
int position = 1;             while (current != NULL)
{
```

```
                cout << position << ". Nama: " << current->nama
<< ", NIM: " << current->nim << endl;
current = current->next;
position++;
            }
        }
    }
};
int main() {
Queue queue;
    queue.enqueue("Fahri Ramadhan", "2311102024");
queue.enqueue("Amanda Syakira", "2311102124");
queue.viewQueue();
    cout << "Jumlah antrian = " << queue.countQueue() <<
endl;     queue.dequeue();     queue.viewQueue();
    cout << "Jumlah antrian = " << queue.countQueue() <<
endl;     queue.clearQueue();     queue.viewQueue();
    cout << "Jumlah antrian = " << queue.countQueue() << endl;
return 0;
}
```

SCREENSHOOT PROGRAM



DESKRIPSI PROGRAM

Program ini menunjukkan bagaimana fungsi antrian mahasiswa dapat digunakan untuk mengatur dan mengelola data antrian menggunakan struktur data linked list. Fungsi-fungsi yang digunakan dalam program ini meliputi fungsi pengecekan, fungsi tambah, fungsi kurangi, fungsi hitung, fungsi hapus, dan fungsi tampilkan.

BAB IV
KESIMPULAN

Setelah melakukan pembelajaran mengenai Queue di Bahasa Pemrograman C++ berikut poin utama yang telah dipelajari :

1. Queue adalah struktur data yang digunakan untuk mengelola data dalam urutan yang spesifik, biasanya dalam urutan yang diterima (First-In-First-Out, FIFO).
2. Queue dapat digunakan untuk mengatur dan mengelola antrian, seperti antrian teller, antrian mahasiswa, dan lain-lain.

3. Fungsi-fungsi Queue seperti isEmpty(), enqueue(), dequeue(), countQueue(), clearQueue(), dan viewQueue() digunakan untuk mengatur dan mengelola data antrian.

DAFTAR PUSTAKA

Kaskus. (2020, 20 Mei) Pengertian Queue Dalam C++. diakses pada 25 Mei 2024 dari https://www.kaskus.co.id/thread/5ec54d20facb95558a5496e1/pengertian-queuedalam-c