

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL VII
QUEUE**



Disusun oleh:
Alfin Ilham Berlianto
2311102047

Dosen pengampu:
Wahyu Andi Saputra, S.Pd., M.Eng.

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

MODUL 7

QUEUE

A. Tujuan

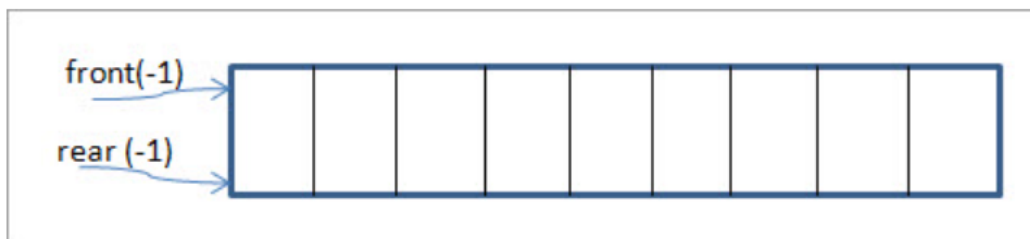
1. Mampu memahami konsep queue pada struktur data dan algoritma.
2. Mampu mengimplementasikan operasi-operasi pada queue.
3. Mampu memecahkan permasalahan dengan sokusi queue.

B. Dasar Teori

Queue adalah struktur data dasar seperti tumpukan. Berbeda dengan stack yang menggunakan pendekatan LIFO, queue menggunakan pendekatan FIFO (first in, first out). Dengan pendekatan ini, item pertama yang ditambahkan ke antrian adalah item pertama yang dikeluarkan dari antrian. Sama seperti Stack, antrian juga merupakan struktur data linier.

Dalam analogi dunia nyata, kita dapat membayangkan antrian bus dimana penumpang menunggu bus dalam antrian atau antrean. Penumpang pertama dalam antrean masuk ke dalam bus terlebih dahulu karena kebetulan penumpang tersebut adalah orang yang datang lebih dulu.

Dalam istilah perangkat lunak, antrian dapat dipandang sebagai sekumpulan atau kumpulan elemen seperti gambar di bawah ini. Elemen-elemennya disusun secara linier.



Disini ada dua ujung yaitu “depan” dan “belakang” antrian. Ketika antrian kosong, maka kedua pointer disetel ke -1.

Penunjuk ujung "belakang" adalah tempat elemen dimasukkan ke dalam antrian. Operasi penambahan/penyisipan elemen ke dalam antrian disebut “enqueue”.

Penunjuk ujung "depan" adalah tempat elemen dihapus dari antrian. Operasi untuk menghapus/menghapus elemen dari antrian disebut “dequeue”.

Bila nilai penunjuk belakang berukuran ke-1, maka dikatakan antrian sudah penuh. Jika bagian depan bernilai null, maka antriannya kosong.

Operasi Dasar

Struktur data antrian mencakup operasi berikut:

- EnQueue: Menambahkan item ke antrian. Penambahan item ke antrian selalu dilakukan di belakang antrian.
- DeQueue: Menghapus item dari antrian. Suatu item selalu dihapus atau dikeluarkan dari antrian dari depan antrian.
- isEmpty: Memeriksa apakah antrian kosong.
- isFull: Memeriksa apakah antrian sudah penuh.
- mengintip: Mendapat elemen di depan antrian tanpa menghapusnya.

Enqueue

Dalam proses ini, langkah-langkah berikut dilakukan:

- Periksa apakah antrian sudah penuh.
- Jika penuh, hasilkan kesalahan luapan dan keluar.
- Jika tidak, tambah 'belakang'.
- Tambahkan elemen ke lokasi yang ditunjuk oleh 'belakang'.
- Kembalikan kesuksesan.

Dequeue

Operasi dequeue terdiri dari langkah-langkah berikut:

- Periksa apakah antriannya kosong.
- Jika kosong, tampilkan kesalahan underflow dan keluar.
- Jika tidak, elemen akses ditunjukkan dengan 'depan'.
- Tambahkan 'depan' untuk menunjuk ke data yang dapat diakses berikutnya.
- Kembalikan kesuksesan.

C. Guided

SOURCE CODE

```
#include <iostream>

using namespace std;

const int maksimalQueue = 5;
int front = 0;
int back = 0;
string queueTeller[maksimalQueue];

bool isFull() {
    return back == maksimalQueue;
}

bool isEmpty() {
    return back == 0;
}

void enqueueAntrian(string data) {
    if(isFull()) {
        cout << "Antrian Penuh" << endl;
    }else {
        queueTeller[back] = data;
        back++;
    }
}

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

int countQueue() {
    return back;
}

void clearQueue() {
    for (int i = 0; i < back; i++)
```

```

    {
        queueTeller[i] = "";
    }

}

void viewQueue() {
    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;
    return 0;
}

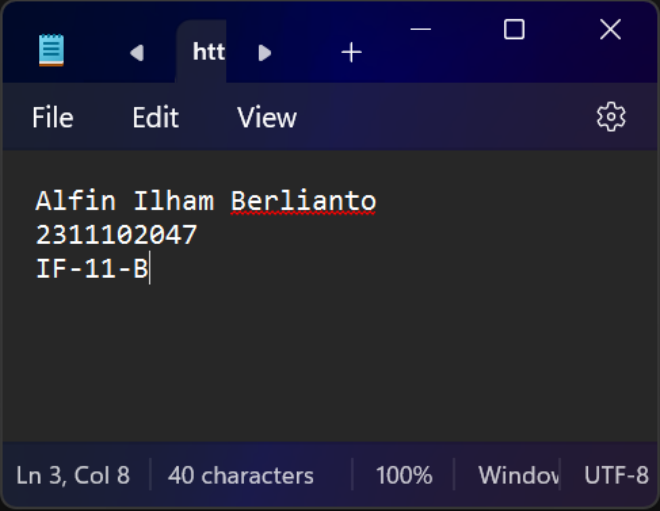
```

SCREENSHOT OUTPUT

```

PS C:\Users\sjsjs\Desktop\code c++\guided 7 1> cd "c:\Users\sjsjs\Desktop\code c++\guided 7 1\" ; if ($?) { g++ main.cpp -o main } ; if ($?) { . \main }
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
PS C:\Users\sjsjs\Desktop\code c++\guided 7 1>

```



DESKRIPSI PROGRAM

Pada guided ini membuat program queue berbasis array, didalam sub program diinisialisasikan maksimal antriannya 5, front dan back nya 0, lalu dibuat fungsi untuk menentukan apakah quenny nya penuh atau kosong, lalu dibuat fungsi apakah queue nya kosong atau tidak, lalu buat void untuk menambah antrian, jika antrian sudah penuh maka akan menampilkan antrian penuh, jika tidak maka akan membuat antrian, lalu dequeue untuk menghapus antrian, jika antriannya kosong maka akan menampilkan bahwa antriannya kosong, jika ada antrian maka antrian akan dihapus. lalu ada fungsi countQueue untuk menghitung berapa antrian yang sudah dibuat, lalu ada void clearQueue untuk menghapus semua antrian yang sudah dibuat. Lalu void viewQueue untuk menampilkan antrian yang sudah dibuat pada enqueue antrian.

D. Unguided

SOURCE CODE

```
#include <iostream>
#include <string>

using namespace std;

struct Data {
    string nama;
    string nim;
    Data *next;
};

int maxData = 5;
Data *front,*back,*newNode,*current,*del;

int countData() {
    if(front== NULL){
        return 0;
    }else {
        int banyakData = 0;
        current = front;
        while(current != NULL) {
            current = current->next;
            banyakData++;
        }
        return banyakData;
    }
}

bool isDataFull() {
    if(countData() == maxData) {
        return true;
    }else {
        return false;
    }
}
```

```

bool isEmpty() {
    if(countData() == 0) {
        return true;
    }else {
        return false;
    }
}

void enqueueData(string nama, string nim) {
    if(isDataFull()) {
        cout << "data penuh!" << endl;
    }else {
        if(isDataEmpty()) {
            front = new Data();
            front->nama = nama;
            front->nim = nim;
            front->next = NULL;
            back = front;
            cout << "Antrian telah ditambahkan!" << endl;
        }else {
            newNode = new Data();
            newNode->nama = nama;
            newNode->nim = nim;
            newNode->next = NULL;
            back->next = newNode;
            back = newNode;
            cout << "Antrian telah ditambahkan!" << endl;
        }
    }
}

void dequeueData() {
    del = front;
    front = front->next;
    del->next = NULL;
    delete del;
    cout << "1 data dihapus" << endl;
}

void clearData() {
    current = front;
    while(current != NULL) {
        newNode = current;
        current = current->next;
        newNode = NULL;
    }
    front = back = NULL;
}

```



```

    }

    void viewData() {
        int i = 1;

        if(isDataEmpty()) {
            cout << "data masih kosong! " <<endl;
        }else {
            current = front;
            cout << "Data Mahasiswa: " <<endl;
            while(current != NULL) {
                cout << i++ << ". " << current->nama << " - " << current->nim <<
endl;
                current = current->next;
            }
        }
    }
}

int main()
{
    int pilih;
    string nama,nim;
    while (true)
    {
        cout << "Data Mahasiswa : " << endl;
        cout << "1. Tambah Data Mahasiswa" << endl;
        cout << "2. Hapus Data Mahasiswa" << endl;
        cout << "3. View Data" << endl;
        cout << "4. Clear Data" << endl;
        cout << "5. Keluar" << endl;
        cout << "Pilih : ";
        cin >> pilih;
        switch (pilih)
        {
            case 1:
                cout << "Masukkan Nama : ";
                cin.ignore();
                getline(cin, nama);
                cout << "Masukkan Nim : ";
                cin >> nim;
                enqueueData(nama,nim);

                break;
            case 2:
                dequeueData();
                break;
            case 3:
                viewData();
                break;

```

```

        case 4:
            clearData();
            cout << "Data dibersihkan " << endl;
            break;
        case 5:
            return 0;
            break;
        default:
            break;
    }
}
return 0;
}

```

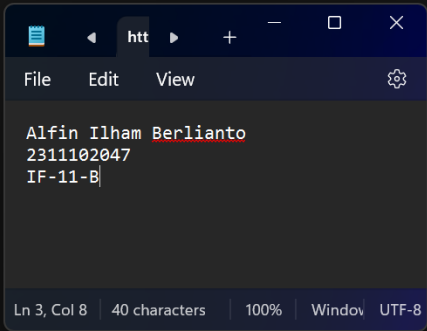
SCREENSHOT OUTPUT

Tambah data antrian

```

PS C:\Users\sjsjs\Desktop\code c++\unguided 7> cd "c:\Users\sjsjs\Desktop\code c++\unguided 7\" ; if ($?) { g++ main.cpp -o m
Data Mahasiswa :
1. Tambah Data Mahasiswa
2. Hapus Data Mahasiswa
3. View Data
4. Clear Data
5. Keluar
Pilih : 1
Masukkan Nama : alfin ilham
Masukkan Nim : 2311102047
Antrian telah ditambahkan!
Data Mahasiswa :
1. Tambah Data Mahasiswa
2. Hapus Data Mahasiswa
3. View Data
4. Clear Data
5. Keluar
Pilih : 

```



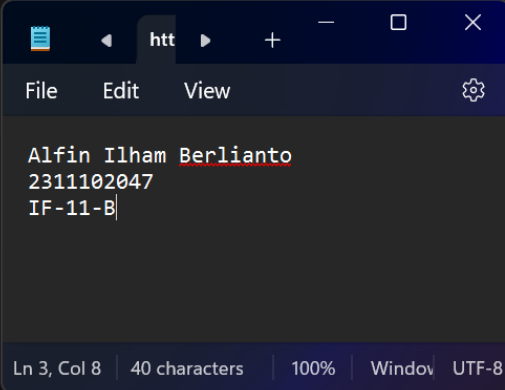
The screenshot shows a Notepad++ window with the following text: Alfin Ilham Berlianto, 2311102047, IF-11-B. The status bar at the bottom indicates 'Ln 3, Col 8', '40 characters', '100%', 'Window', and 'UTF-8'.

Hapus data antrian

```

Pilih : 1
Masukkan Nama : raka andrik
Masukkan Nim : 2311102057
Antrian telah ditambahkan!
Data Mahasiswa :
1. Tambah Data Mahasiswa
2. Hapus Data Mahasiswa
3. View Data
4. Clear Data
5. Keluar
Pilih : 2
1 data dihapus
Data Mahasiswa :
1. Tambah Data Mahasiswa
2. Hapus Data Mahasiswa
3. View Data
4. Clear Data
5. Keluar
Pilih : 

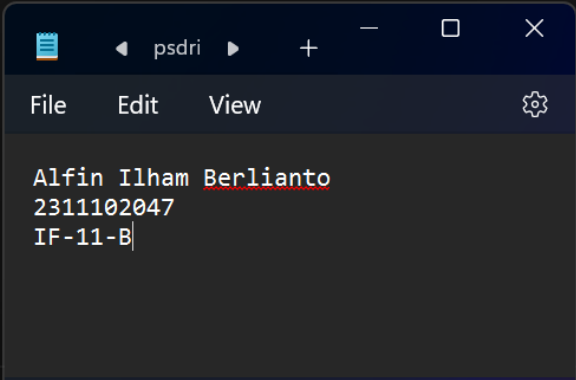
```



The screenshot shows a Notepad++ window with the same text as the previous screenshot: Alfin Ilham Berlianto, 2311102047, IF-11-B. The status bar at the bottom indicates 'Ln 3, Col 8', '40 characters', '100%', 'Window', and 'UTF-8'.

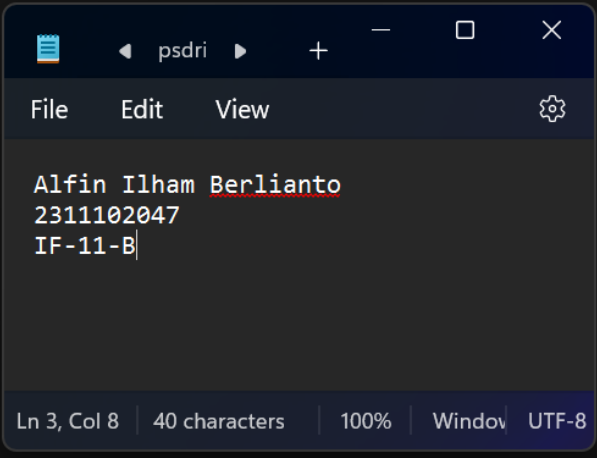
Tampilkan data antrian

```
1 data dihapus
Data Mahasiswa :
1. Tambah Data Mahasiswa
2. Hapus Data Mahasiswa
3. View Data
4. Clear Data
5. Keluar
Pilih : 3
Data Mahasiswa:
1. raka andrik - 2311102057
Data Mahasiswa :
1. Tambah Data Mahasiswa
2. Hapus Data Mahasiswa
3. View Data
4. Clear Data
5. Keluar
Pilih : 
```



Clear data mahasiswa

```
5. Keluar
Pilih : 4
Data dibersihkan
Data Mahasiswa :
1. Tambah Data Mahasiswa
2. Hapus Data Mahasiswa
3. View Data
4. Clear Data
5. Keluar
Pilih : 3
data masih kosong!
Data Mahasiswa :
1. Tambah Data Mahasiswa
2. Hapus Data Mahasiswa
3. View Data
4. Clear Data
5. Keluar
Pilih : 
```



DESKRIPSI PROGRAM

Pada unguided ini membuat program queue berbasis linked list, pada unguided ini data queue-nya yaitu nama dan nim mahasiswa, di sub program diinisialisasikan batas antriannya 5, terdapat pointer front, back, newNode, current, del. Pertama di subprogram terdapat fungsi untuk menghitung data antrian dengan void countData, lalu ada fungsi isDataFull untuk menentukan apakah data antriannya penuh atau tidak, lalu ada fungsi isDataEmpty untuk menentukan apakah data antriannya kosong atau tidak, lalu untuk menambahkan data antrian menggunakan void enqueueData, untuk menghapus data antrian menggunakan void dequeueData, untuk membersihkan data menggunakan void clearData, untuk menampilkan data antrian menggunakan void viewData.

DAFTAR PUSTAKA

1. Tutorialspoint (2023, 2 September) Program C++ untuk Mengimplementasikan Antrian Berbasis Array. Diakses pada 28 Mei 2024, dari <https://g.co/kgs/6AYgy4M>
- 2, Software Testing Help, (2023, 23 Desember) Struktur Data Antrian Dalam C++ Dengan Ilustrasi. Diakses pada 19 Mei 2024, dari <https://www.nblognlife.com/2014/04/stack-pada-c.html>