

**LAPORAN PRAKTIKUM STRUKTUR
DATA DAN ALGORITMA**

**MODUL VII
QUEUE**



Disusun Oleh :
Muhammad Arsyad Zaidan (2311102058)

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

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

B. Dasar Teori

Pengertian Queue

Queue atau dalam bahasa Indonesia yang berarti antrean adalah struktur data yang menyusun elemen-elemen data dalam urutan linier. Prinsip dasar dari struktur data ini adalah “First In, First Out” (FIFO) yang berarti elemen data yang pertama dimasukkan ke dalam antrean akan menjadi yang pertama pula untuk dikeluarkan. Queue memiliki peran yang penting dalam berbagai aplikasi dan algoritma. Salah satu fungsi utamanya adalah mengatur dan mengelola antrean tugas atau operasi secara efisien. Dalam sistem komputasi, ia digunakan untuk menangani tugas-tugas seperti penjadwalan proses, antrean pesan, dan manajemen sumber daya.

Prinsip FIFO pada Queue

Caranya bekerja adalah seperti jejeran orang yang sedang menunggu antrean di supermarket di mana orang pertama yang datang adalah yang pertama dilayani (First In, First Out). Pada struktur data ini, urutan pertama (data yang akan dikeluarkan) disebut Front atau Head. Sebaliknya, data pada urutan terakhir (data yang baru saja ditambahkan) disebut Back, Rear, atau Tail. Proses untuk menambahkan data pada antrean disebut dengan Enqueue, sedangkan proses untuk menghapus data dari antrean disebut dengan Dequeue.



Gambar 1.1 Queue Data Structure

Perbedaan antara stack dan queue terdapat pada aturan penambahan dan penghapusan elemen. Pada stack, operasi penambahan dan penghapusan elemen dilakukan di satu ujung. Elemen yang terakhir diinputkan akan berada paling dengan dengan ujung atau dianggap paling atas sehingga pada operasi penghapusan, elemen teratas tersebut akan dihapus paling awal, sifat demikian dikenal dengan LIFO.

Pada Queue, operasi tersebut dilakukan ditempat berbeda (melalui salah satu ujung) karena perubahan data selalu mengacu pada Head, maka hanya ada 1 jenis insert maupun delete. Prosedur ini sering disebut Enqueue dan Dequeue pada kasus Queue. Untuk Enqueue, cukup tambahkan elemen setelah elemen terakhir Queue, dan untuk Dequeue, cukup "geser"kan Head menjadi elemen selanjutnya.

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.

1. Latihan Guided dan Unguided

1) Guided

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
            { // Antriannya 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;
}

```

Output Code

```

PS C:\Learning Code\STRUKDAT (PRAKTEK)\New folder> & 'c:\U
sDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-jyxmbzpj
py1ysz1.0ir' '--pid=Microsoft-MIEngine-Pid-k32lckyf.nql' '-
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:\Learning Code\STRUKDAT (PRAKTEK)\New folder>

```

Nama : Muhammad Arsyad Zaidan
 Kelas : IF-11-B
 NIM : 2311102058

Deskripsi Code

Program ini mengimplementasikan antrian (queue) sederhana menggunakan array dan terdapat fungsi-fungsi yang digunakan dalam program yaitu,

- 1) `isFull()` digunakan untuk mengecek apakah antrian penuh.
- 2) `isEmpty()` yang berfungsi untuk mengecek apakah antrian kosong.
- 3) `enqueueAntrian(string data)` yang berfungsi untuk menambahkan elemen ke dalam antrian jika belum penuh.
- 4) `dequeueAntrian()` yang berfungsi untuk mengeluarkan elemen dari antrian jika tidak kosong, dan menggeser elemen lainnya maju satu posisi.

- 5) `countQueue()` yang berfungsi menghitung jumlah elemen dalam antrian.
- 6) `clearQueue()` yang berfungsi menghapus semua elemen dalam antrian.
- 7) `viewQueue()` yang berfungsi untuk menampilkan semua elemen dalam antrian beserta posisinya.

Pada fungsi `main()`, antrian diisi dengan beberapa nama, ditampilkan, dihitung jumlahnya, satu elemen dihapus, antrian ditampilkan kembali, dan akhirnya semua elemen dihapus serta antrian ditampilkan untuk terakhir kalinya.

2) Unguided

Unguided1

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;

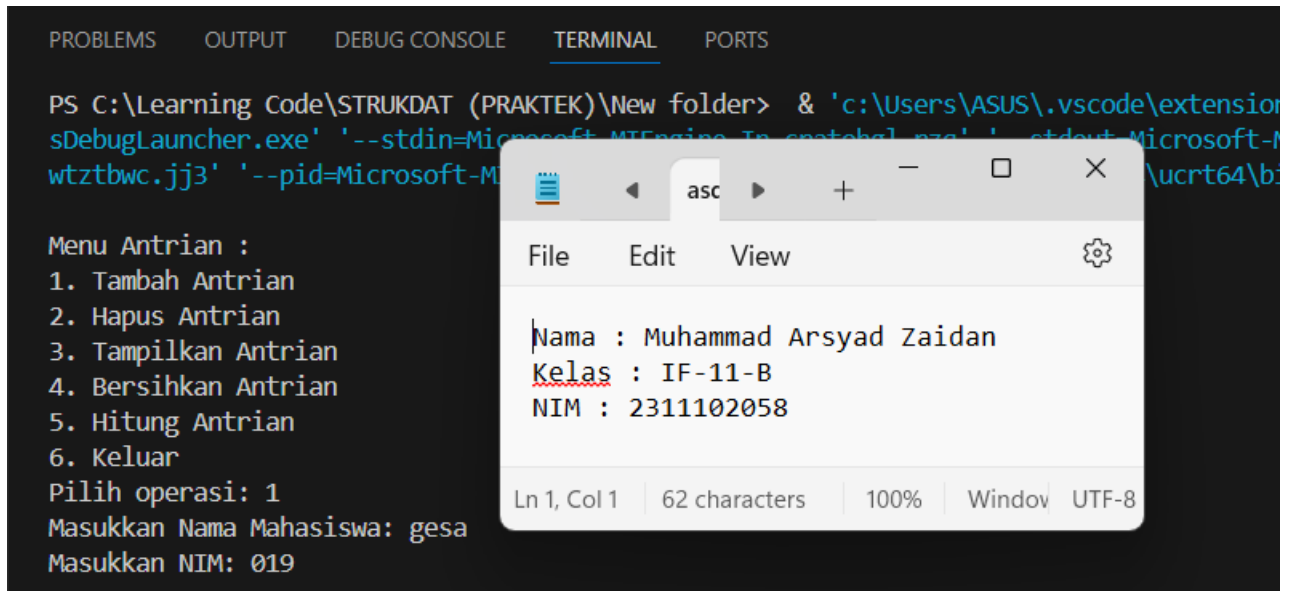
    do {
        displayMenu();
        cin >> choice;
        switch (choice) {
            case 1:
                cout << "Masukkan Nama Mahasiswa: ";
                cin.ignore(); // Mengabaikan newline yang tersisa di input buffer
                getline(cin, namaMahasiswa);
                cout << "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;
}
```

Output Code



```
PS C:\Learning Code\STRUKDAT (PRAKTEK)\New folder> & 'c:\Users\ASUS\.vscode\extension
sDebuglauncher.exe' '--stdin=Microsoft-MTEngine-In-Engine\pza' --stdout=Microsoft-M
wtztbwc.jj3' '--pid=Microsoft-M
```

Menu Antrian :

1. Tambah Antrian
2. Hapus Antrian
3. Tampilkan Antrian
4. Bersihkan Antrian
5. Hitung Antrian
6. Keluar

Pilih operasi: 1
Masukkan Nama Mahasiswa: gesa
Masukkan NIM: 019

Nama : Muhammad Arsyad Zaidan
Kelas : IF-11-B
NIM : 2311102058

Deskripsi Code

Node yang tertuju pada mahasiswa untuk direpresentasikan pada linked list, yang mana berfungsi untuk menyimpan data mahasiswa seperti nama dan NIM yang disertakan pointer ke node berikutnya. Program linked list ini mengimplementasikan antrian (queue) yang dapat menambah, menghapus, menampilkan, membersihkan, dan menghitung pada data dalam pemrograman.

Unguided2

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;
}

```

Output Code

1) Menambah Antrian pada Output

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Learning Code\STRUKDAT (PRAKTEK)\New folder> & 'c:\Users\ASUS\.vscode\extension  
sDebugLauncher.exe' '--stdin=Microsoft-MIEngine-In-hzfv5jwy.s2k' '--stdout=Microsoft-f  
g4ox00c.zwy' '--pid=Microsoft-MIEngine-Pid-pzuspqfe.ugo' '--dbgExe=C:\msys64\ucrt64\b
```

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: daffa

Masukkan NIM: 020

Menu Antrian Mahasiswa:

1. Tambah Antrian

2. Hapus Antrian

3. Tampilkan Antrian

4. Bersihkan Antrian

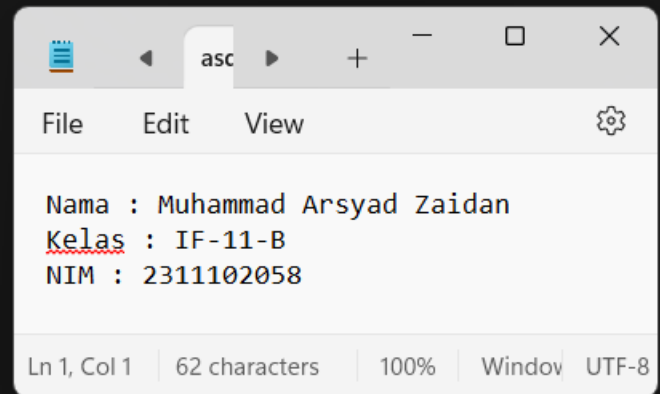
5. Hitung Antrian

6. Keluar

Pilih operasi: 1

Masukkan Nama Mahasiswa: gesa

Masukkan NIM: 019











```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

Pilih operasi: 1
Masukkan Nama Mahasiswa: gesa
Masukkan NIM: 019

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: hamzah
Masukkan NIM: 021

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: jaidan
Masukkan NIM: 022
```

asc

File Edit View 

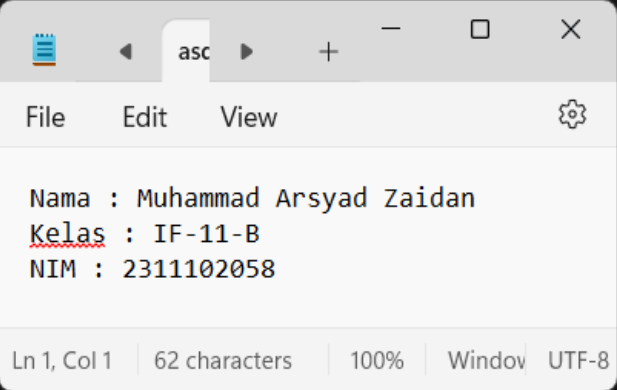
Nama : Muhammad Arsyad Zaidan
Kelas : IF-11-B
NIM : 2311102058

Ln 1, Col 1 62 characters 100% Window UTF-8

2) Menghapus Antrian pada Output


```
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

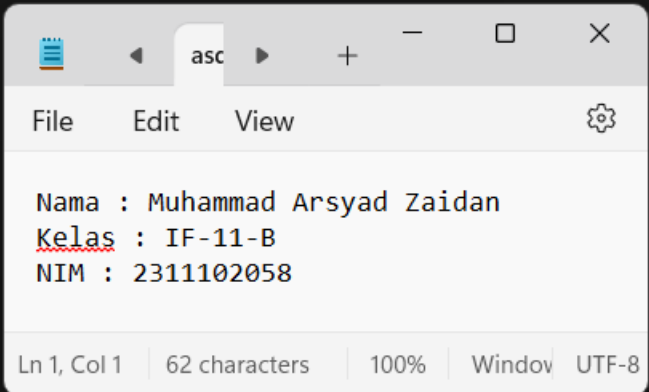
Menu Antrian Mahasiswa:
1. Tambah Antrian
2. Hapus Antrian
3. Tampilkan Antrian
4. Bersihkan Antrian
5. Hitung Antrian
6. Keluar
Pilih operasi: 3
Data antrian mahasiswa:
1. Nama: gesa, NIM: 019
2. Nama: hamzah, NIM: 021
3. Nama: jaidan, NIM: 022
```



A screenshot of a Notepad window with a single tab labeled 'asc'. The window contains the following text: 'Nama : Muhammad Arsyad Zaidan', 'Kelas : IF-11-B', and 'NIM : 2311102058'. The status bar at the bottom indicates 'Ln 1, Col 1', '62 characters', '100%', 'Window', and 'UTF-8'.

3) Menampilkan Antrian pada Output

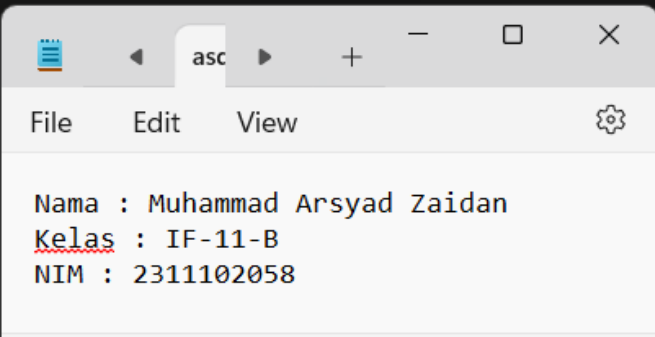
```
Menu Antrian Mahasiswa:
1. Tambah Antrian
2. Hapus Antrian
3. Tampilkan Antrian
4. Bersihkan Antrian
5. Hitung Antrian
6. Keluar
Pilih operasi: 3
Data antrian mahasiswa:
1. Nama: gesa, NIM: 019
2. Nama: hamzah, NIM: 021
3. Nama: jaidan, NIM: 022
```



A screenshot of a Notepad window with a single tab labeled 'asc'. The window contains the following text: 'Nama : Muhammad Arsyad Zaidan', 'Kelas : IF-11-B', and 'NIM : 2311102058'. The status bar at the bottom indicates 'Ln 1, Col 1', '62 characters', '100%', 'Window', and 'UTF-8'.

4) Menghitung Antrian pada Output

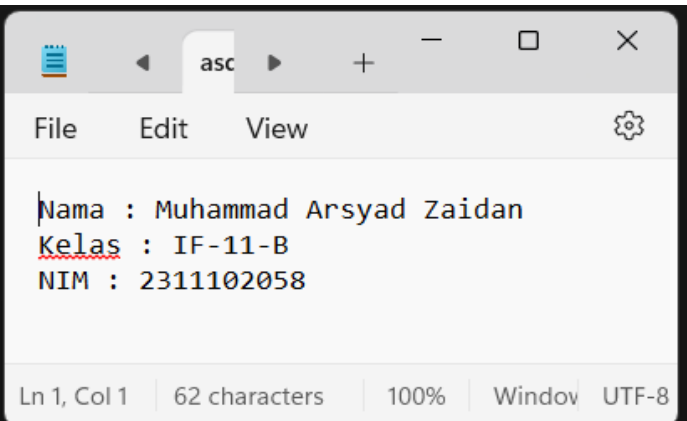
```
Menu Antrian Mahasiswa:
1. Tambah Antrian
2. Hapus Antrian
3. Tampilkan Antrian
4. Bersihkan Antrian
5. Hitung Antrian
6. Keluar
Pilih operasi: 5
Jumlah antrian = 3
```



A screenshot of a Notepad window with a single tab labeled 'asc'. The window contains the following text: 'Nama : Muhammad Arsyad Zaidan', 'Kelas : IF-11-B', and 'NIM : 2311102058'. The status bar at the bottom indicates 'Ln 1, Col 1', '62 characters', '100%', 'Window', and 'UTF-8'.

5) Membersihkan Antrian pada Output

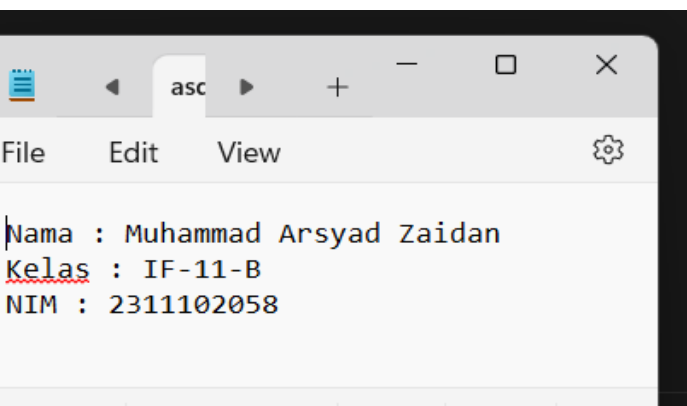
```
Menu Antrian Mahasiswa:
1. Tambah Antrian
2. Hapus Antrian
3. Tampilkan Antrian
4. Bersihkan Antrian
5. Hitung Antrian
6. Keluar
Pilih operasi: 4
Antrian Berhasil Dihapus
Antrian Berhasil Dihapus
Antrian Berhasil Dihapus
```



The screenshot shows a code editor window with a menu and user input. The menu lists six options: 1. Tambah Antrian, 2. Hapus Antrian, 3. Tampilkan Antrian, 4. Bersihkan Antrian, 5. Hitung Antrian, and 6. Keluar. The user has selected option 4, and the program has responded with 'Antrian Berhasil Dihapus' three times. The code editor window shows the following text: 'Nama : Muhammad Arsyad Zaidan', 'Kelas : IF-11-B', and 'NIM : 2311102058'. The status bar at the bottom indicates 'Ln 1, Col 1', '62 characters', '100%', 'Window', and 'UTF-8'.

6) Keluar Antrian

```
Menu Antrian Mahasiswa:
1. Tambah Antrian
2. Hapus Antrian
3. Tampilkan Antrian
4. Bersihkan Antrian
5. Hitung Antrian
6. Keluar
Pilih operasi: 6
Keluar dari program.
PS C:\Learning Code\STRUKE
```



The screenshot shows a code editor window with a menu and user input. The menu lists six options: 1. Tambah Antrian, 2. Hapus Antrian, 3. Tampilkan Antrian, 4. Bersihkan Antrian, 5. Hitung Antrian, and 6. Keluar. The user has selected option 6, and the program has responded with 'Keluar dari program.'. The code editor window shows the following text: 'Nama : Muhammad Arsyad Zaidan', 'Kelas : IF-11-B', and 'NIM : 2311102058'. The status bar at the bottom indicates 'Ln 1, Col 1', '62 characters', '100%', 'Window', and 'UTF-8'.

Deskripsi Code

Code yang sama seperti pada soal sebelumnya, program tersebut dapat menambah, menghapus, menampilkan, membersihkan, menghitung, hingga keluar dari program. Program tersebut tidak jauh dari pengimplementasian dari antrian (queue) yang bertugas untuk monitoring data dari mahasiswa.

2. Kesimpulan

Dari pratikum ini, mahasiswa dapat mempelajari mengimplementasikan queue. Queue (antrian) adalah barisan elemen yang apabila elemen ditambah maka penambahannya berada di posisi belakang (rear) dan jika dilakukan pengambilan elemen dilakukan di elemen paling depan (front). Oleh karena itu, queue bersifat FIFO (first in first out).

3. Referensi

[1] Tim Asisten Asprak, "Queue", Learning Management System, 2024.

- [2] Maulana, R. (2023, November 29). "Struktur Data Queue: Pengertian, Fungsi, dan Jenisnya". Dicoding Blog. <https://www.dicoding.com/blog/struktur-data-queue-pengertian-fungsi-dan-jenisnya/>. Terakhir kali diakses pada tanggal 24 Mei 2024