

# LAPORAN PRAKTIKUM

## MODUL 8

### QUEUE



Nama :

Fajar Budiawan ( 2311104039 )

Dosen :

Yudha Islami Sulistya S.Kom,  
M.Cs

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

FAKULTAS INFORMATIKA

**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## Unguided

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

```
1  #include <iostream>
2  using namespace std;
3
4  // Definisikan struktur Node untuk linked list
5  struct Node {
6      string data;    // Data untuk disimpan (bisa tipe lain, di sini string)
7      Node* next;     // Pointer ke node berikutnya dalam list
8  };
9
10 // Definisikan kelas Queue untuk mengelola antrian berbasis linked list
11 class Queue {
12 private:
13     Node* front;    // Pointer ke depan antrian
14     Node* back;     // Pointer ke belakang antrian
15
16 public:
17     // Konstruktor, inisialisasi queue dengan front dan back null
18     Queue() : front(nullptr), back(nullptr) {}
19
20     // Cek apakah antrian kosong
21     bool isEmpty() {
22         return front == nullptr;
23     }
24
25     // Menambahkan elemen ke dalam antrian (enqueue)
26     void enqueue(const string& data) {
27         Node* newNode = new Node{data, nullptr}; // Buat node baru
28
29         if (isEmpty()) {
30             front = back = newNode; // Jika kosong, front dan back menunjuk ke node baru
31         } else {
32             back->next = newNode; // Sambungkan node baru di belakang antrian
33             back = newNode;      // Update pointer back ke node baru
34         }
35     }
36
37     // Menghapus elemen dari antrian (dequeue)
38     void dequeue() {
39         if (isEmpty()) {
40             cout << "Antrian Kosong" << endl;
41             return;
42         }
43
44         Node* temp = front; // Simpan node depan sementara
45         front = front->next; // Pindahkan front ke node berikutnya
46         if (front == nullptr) {
47             back = nullptr; // Jika antrian kosong, back juga null
48         }
49         delete temp; // Hapus node lama (node yang terdepan)
50     }
51
52     // Menghitung jumlah elemen dalam antrian
53     int count() {
54         int count = 0;
55         Node* current = front;
56         while (current != nullptr) {
57             count++;
58             current = current->next;
59         }
60         return count;
61     }
62
63     // Menghapus semua elemen dalam antrian
64     void clear() {
65         while (!isEmpty()) {
66             dequeue(); // Dequeue hingga antrian kosong
67         }
68     }
69
70     // Menampilkan elemen-elemen dalam antrian
71     void view() {
72         if (isEmpty()) {
73             cout << "Antrian Kosong" << endl;
74             return;
75         }
76
77         cout << "Data antrian teller:" << endl;
78         Node* current = front;
79         int position = 1;
80         while (current != nullptr) {
81             cout << position << ". " << current->data << endl;
82             current = current->next;
83             position++;
84         }
85     }
86 };
87
88 int main() {
89     Queue q; // Membuat objek Queue
90
91     // Menambahkan elemen ke antrian
92     q.enqueue("Andi");
93     q.enqueue("Maya");
94     q.view(); // Menampilkan antrian
95     cout << "Jumlah Antrian = " << q.count() << endl;
96
97     // Menghapus elemen dari antrian
98     q.dequeue();
99     q.view(); // Menampilkan antrian setelah dequeue
100    cout << "Jumlah Antrian = " << q.count() << endl;
101
102    // Menghapus semua elemen dalam antrian
103    q.clear();
104    q.view(); // Menampilkan antrian setelah clear
105    cout << "Jumlah Antrian = " << q.count() << endl;
106
107    return 0;
108 }
109
```

Output

```
PS D:\Praktikum Struktur
n_8\Ungudied\" ; if ($?)
Data antrian teller:
1. Andi
2. Maya
Jumlah Antrian = 2
Data antrian teller:
1. Maya
Jumlah Antrian = 1
Antrian Kosong
Jumlah Antrian = 0
PS D:\Praktikum Struktur
```

2. Dari nomor 1 buatlah konsep antri dengan atribut Nama mahasiswa dan NIM Mahasiswa

```
3. #include <iostream>
4. using namespace std;
5.
6. // Definisikan struktur Node untuk linked list
7. struct Node {
8.     string nama; // Nama mahasiswa
9.     string nim;  // NIM mahasiswa
10.    Node* next;   // Pointer ke node berikutnya dalam list
11.};
12.
13.// Definisikan kelas Queue untuk mengelola antrian berbasis linked list
14.class Queue {
15.private:
16.    Node* front; // Pointer ke depan antrian
17.    Node* back;  // Pointer ke belakang antrian
18.
19.public:
20.    // Konstruktor, inisialisasi queue dengan front dan back null
21.    Queue() : front(nullptr), back(nullptr) {}
22.
23.    // Cek apakah antrian kosong
24.    bool isEmpty() {
25.        return front == nullptr;
26.    }
27.
28.    // Menambahkan elemen ke dalam antrian (enqueue)
29.    void enqueue(const string& nama, const string& nim) {
30.        Node* newNode = new Node{nama, nim, nullptr}; // Buat node baru
31.
32.        if (isEmpty()) {
33.            front = back = newNode; // Jika kosong, front dan back
menunjuk ke node baru
34.        } else {
35.            back->next = newNode; // Sambungkan node baru di belakang
antrian
36.            back = newNode;      // Update pointer back ke node baru
37.        }
38.    }
```

```

39.
40. // Menghapus elemen dari antrian (dequeue)
41. void dequeue() {
42.     if (isEmpty()) {
43.         cout << "Antrian Kosong" << endl;
44.         return;
45.     }
46.
47.     Node* temp = front; // Simpan node depan sementara
48.     front = front->next; // Pindahkan front ke node berikutnya
49.     if (front == nullptr) {
50.         back = nullptr; // Jika antrian kosong, back juga null
51.     }
52.     delete temp; // Hapus node lama (node yang terdepan)
53. }
54.
55. // Menghitung jumlah elemen dalam antrian
56. int count() {
57.     int count = 0;
58.     Node* current = front;
59.     while (current != nullptr) {
60.         count++;
61.         current = current->next;
62.     }
63.     return count;
64. }
65.
66. // Menghapus semua elemen dalam antrian
67. void clear() {
68.     while (!isEmpty()) {
69.         dequeue(); // Dequeue hingga antrian kosong
70.     }
71. }
72.
73. // Menampilkan elemen-elemen dalam antrian
74. void view() {
75.     if (isEmpty()) {
76.         cout << "Antrian Kosong" << endl;
77.         return;
78.     }
79.
80.     cout << "Data Antrian Mahasiswa:" << endl;
81.     Node* current = front;
82.     int position = 1;
83.     while (current != nullptr) {
84.         cout << position << ". Nama: " << current->nama << ", NIM: "
85.         << current->nim << endl;
86.         current = current->next;
87.         position++;
88.     }
89. };

```

```

90.
91. int main() {
92.     Queue q; // Membuat objek Queue
93.     int pilihan;
94.     string nama, nim;
95.
96.     do {
97.         cout << "\nMenu Antrian Mahasiswa:" << endl;
98.         cout << "1. Tambah Mahasiswa ke Antrian (Enqueue)" << endl;
99.         cout << "2. Hapus Mahasiswa dari Antrian (Dequeue)" << endl;
100.        cout << "3. Lihat Daftar Antrian Mahasiswa" << endl;
101.        cout << "4. Hapus Semua Antrian" << endl;
102.        cout << "5. Keluar" << endl;
103.        cout << "Pilih opsi: ";
104.        cin >> pilihan;
105.
106.        switch (pilihan) {
107.            case 1:
108.                cout << "Masukkan Nama Mahasiswa: ";
109.                cin.ignore(); // Mengabaikan newline tersisa dari
                input sebelumnya
110.                getline(cin, nama);
111.                cout << "Masukkan NIM Mahasiswa: ";
112.                getline(cin, nim);
113.                q.enqueue(nama, nim);
114.                cout << "Mahasiswa berhasil ditambahkan ke
                antrian." << endl;
115.                break;
116.
117.            case 2:
118.                q.dequeue();
119.                cout << "Mahasiswa di depan antrian telah
                dihapus." << endl;
120.                break;
121.
122.            case 3:
123.                q.view();
124.                cout << "Jumlah Antrian = " << q.count() << endl;
125.                break;
126.
127.            case 4:
128.                q.clear();
129.                cout << "Semua antrian telah dihapus." << endl;
130.                break;
131.
132.            case 5:
133.                cout << "Keluar dari program." << endl;
134.                break;
135.
136.            default:
137.                cout << "Pilihan tidak valid. Silakan coba lagi."
                << endl;

```

```

138.         }
139.     } while (pilihan != 5);
140.
141.     return 0;
142. }
143.

```

#### Output

```

Menu Antrian Mahasiswa:
1. Tambah Mahasiswa ke Antrian (Enqueue)
2. Hapus Mahasiswa dari Antrian (Dequeue)
3. Lihat Daftar Antrian Mahasiswa
4. Hapus Semua Antrian
5. Keluar
Pilih opsi: 1
Masukkan Nama Mahasiswa: jidan
Masukkan NIM Mahasiswa: 232323
Mahasiswa berhasil ditambahkan ke antrian.

Menu Antrian Mahasiswa:
1. Tambah Mahasiswa ke Antrian (Enqueue)
2. Hapus Mahasiswa dari Antrian (Dequeue)
3. Lihat Daftar Antrian Mahasiswa
4. Hapus Semua Antrian
5. Keluar
Pilih opsi: 5
Keluar dari program.
PS D:\Praktikum Struktur Data\08_Pengenalan_CPP_Bagian_8\Ungudied>

```

3. Modifikasi program pada soal 1 sehingga mahasiswa dapat diprioritaskan berdasarkan NIM (NIM yang lebih kecil didahulukan pada saat output).

```

4. #include <iostream>
5. using namespace std;
6.
7. // Definisikan struktur Node untuk linked list
8. struct Node {
9.     string nama; // Nama mahasiswa
10.    string nim; // NIM mahasiswa
11.    Node* next; // Pointer ke node berikutnya dalam list
12.};
13.
14.// Definisikan kelas Queue untuk mengelola antrian berbasis linked list
15.class Queue {
16.private:
17.    Node* front; // Pointer ke depan antrian
18.    Node* back; // Pointer ke belakang antrian
19.
20.public:
21.    // Konstruktor, inisialisasi queue dengan front dan back null
22.    Queue() : front(nullptr), back(nullptr) {}
23.
24.    // Cek apakah antrian kosong
25.    bool isEmpty() {
26.        return front == nullptr;

```

```

27.     }
28.
29.     // Menambahkan elemen ke dalam antrian (enqueue)
30.     void enqueue(const string& nama, const string& nim) {
31.         Node* newNode = new Node{nama, nim, nullptr}; // Buat node baru
32.
33.         if (isEmpty()) {
34.             front = back = newNode; // Jika kosong, front dan back
menunjuk ke node baru
35.         } else {
36.             back->next = newNode; // Sambungkan node baru di belakang
antrian
37.             back = newNode;      // Update pointer back ke node baru
38.         }
39.     }
40.
41.     // Menghapus elemen dari antrian (dequeue)
42.     void dequeue() {
43.         if (isEmpty()) {
44.             cout << "Antrian Kosong" << endl;
45.             return;
46.         }
47.
48.         Node* temp = front; // Simpan node depan sementara
49.         front = front->next; // Pindahkan front ke node berikutnya
50.         if (front == nullptr) {
51.             back = nullptr; // Jika antrian kosong, back juga null
52.         }
53.         delete temp; // Hapus node lama (node yang terdepan)
54.     }
55.
56.     // Menghitung jumlah elemen dalam antrian
57.     int count() {
58.         int count = 0;
59.         Node* current = front;
60.         while (current != nullptr) {
61.             count++;
62.             current = current->next;
63.         }
64.         return count;
65.     }
66.
67.     // Menghapus semua elemen dalam antrian
68.     void clear() {
69.         while (!isEmpty()) {
70.             dequeue(); // Dequeue hingga antrian kosong
71.         }
72.     }
73.
74.     // Menampilkan elemen-elemen dalam antrian
75.     void view() {
76.         if (isEmpty()) {

```

```

77.         cout << "Antrian Kosong" << endl;
78.         return;
79.     }
80.
81.     cout << "Data Antrian Mahasiswa:" << endl;
82.     Node* current = front;
83.     int position = 1;
84.     while (current != nullptr) {
85.         cout << position << ". Nama: " << current->nama << ", NIM: "
86.         << current->nim << endl;
87.         current = current->next;
88.         position++;
89.     }
90. };
91.
92. int main() {
93.     Queue q; // Membuat objek Queue
94.     int pilihan;
95.     string nama, nim;
96.
97.     do {
98.         cout << "\nMenu Antrian Mahasiswa:" << endl;
99.         cout << "1. Tambah Mahasiswa ke Antrian (Enqueue)" << endl;
100.        cout << "2. Hapus Mahasiswa dari Antrian (Dequeue)" <<
101.        endl;
102.        cout << "3. Lihat Daftar Antrian Mahasiswa" << endl;
103.        cout << "4. Hapus Semua Antrian" << endl;
104.        cout << "5. Keluar" << endl;
105.        cout << "Pilih opsi: ";
106.        cin >> pilihan;
107.
108.        switch (pilihan) {
109.            case 1:
110.                cout << "Masukkan Nama Mahasiswa: ";
111.                cin.ignore(); // Mengabaikan newline tersisa dari
112.                input sebelumnya
113.                getline(cin, nama);
114.                cout << "Masukkan NIM Mahasiswa: ";
115.                getline(cin, nim);
116.                q.enqueue(nama, nim);
117.                cout << "Mahasiswa berhasil ditambahkan ke
118.                antrian." << endl;
119.                break;
120.
121.            case 2:
122.                q.dequeue();
123.                cout << "Mahasiswa di depan antrian telah
124.                dihapus." << endl;
125.                break;
126.
127.            case 3:

```



```

124.             q.view();
125.             cout << "Jumlah Antrian = " << q.count() << endl;
126.             break;
127.
128.             case 4:
129.                 q.clear();
130.                 cout << "Semua antrian telah dihapus." << endl;
131.                 break;
132.
133.             case 5:
134.                 cout << "Keluar dari program." << endl;
135.                 break;
136.
137.             default:
138.                 cout << "Pilihan tidak valid. Silakan coba lagi."
139.                 << endl;
140.             }
141.         } while (pilihan != 5);
142.         return 0;
143.     }
144.

```

Output

```

Menu Antrian Mahasiswa:
1. Tambah Mahasiswa ke Antrian (Enqueue)
2. Hapus Mahasiswa dari Antrian (Dequeue)
3. Lihat Daftar Antrian Mahasiswa
4. Hapus Semua Antrian
5. Keluar
Pilih opsi: 1
Masukkan Nama Mahasiswa: jidan
Masukkan NIM Mahasiswa: 232323
Mahasiswa berhasil ditambahkan ke antrian.

Menu Antrian Mahasiswa:
1. Tambah Mahasiswa ke Antrian (Enqueue)
2. Hapus Mahasiswa dari Antrian (Dequeue)
3. Lihat Daftar Antrian Mahasiswa
4. Hapus Semua Antrian
5. Keluar
Pilih opsi: 5
Keluar dari program.
PS D:\Praktikum Struktur Data\08_Pengenalan_CPP_Bagian_8\Ungudied>

```