STIKOM BALI

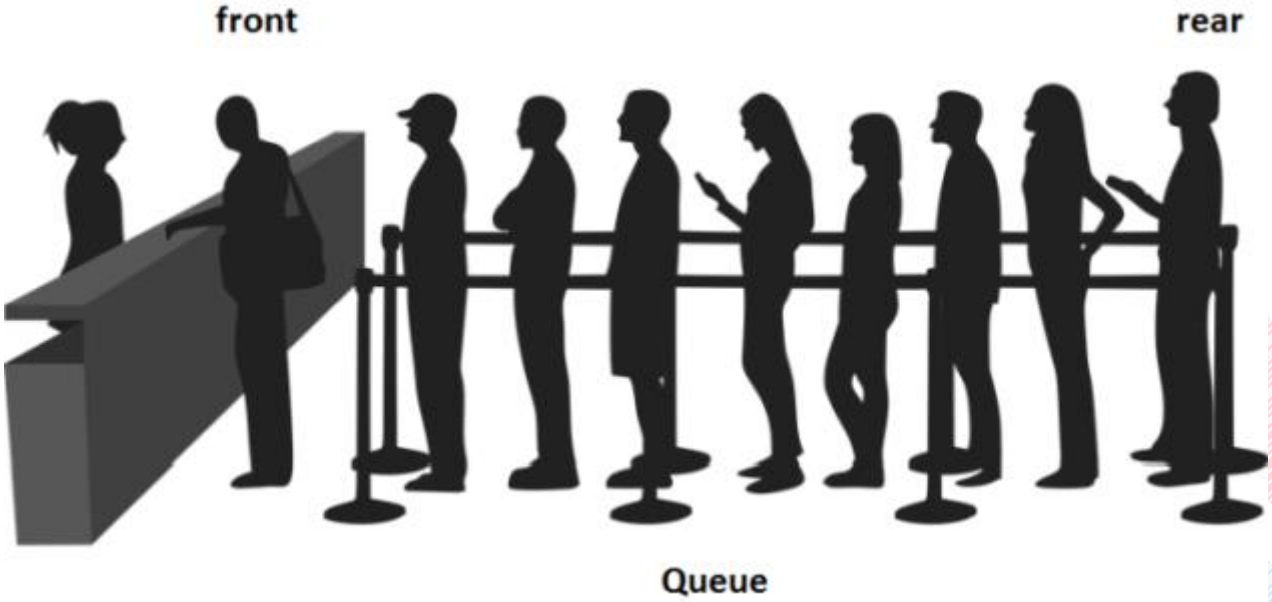
QUEUE/ANTRIAN

Ni Kadek Sumiari, S.Kom., M.MSI

Pengertian

Merupakan sekumpulan data yang mengalami penambahan data(elemen) melalui satu sisi, yaitu depan (head) dan penghapusan data(elemen) melalui sisi belakang (tail)

Gambaran Konsep Queue



- Pada Queue berlaku konsep **FIFO (First In First Out)**, yaitu data yang pertama masuk akan keluar terlebih dahulu. Dan data yang terakhir masuk akan keluar paling akhir

Operasi dalam Queue

- IsFull
  - Menyatakan bahwa antrian sudah penuh
- IsEmpty
  - Menyatakan bahwa antrian masih kosong
- Enqueue
  - Melakukan input ke antrian
- Dequeue
  - Mengeluarkan sebuah nilai pada antrian
- Print
  - Menampilkan isi dari antrian
- Clear
  - Mengembalikan posisi antrian kosong

Program QUEUE

- Untuk mengimplementasikan program queue di C++ diperlukan tiga method atau fungsi yaitu **`enqueue();`** untuk menambahkan data ke antrian, **`dequeue();`** untuk me ngeluarkan data dari antrian dan **`printQueue()`** untuk menampilkan queue.
- Selain tiga fungsi tersebut perlu dibuat dua fungsi opsional untuk mengecek apakah antrian kosong **`isEmpty()`** dan antrian penuh **`isFull().`**
- Untuk menyimpan data bisa menggunakan empty array dengan maksimum array yang nanti akan definisikan sebagai maksimum antrian, jadi bisa diketahui indeks pertama adalah *front* dan data indeks yang kosong untuk menambahkan data sebagai *rear*-nya.
- Untuk data antriannya terstruktur bisa menggunakan *struct* sehingga lebih mudah mengakses data *front*, *rear* dan *array* datanya sendiri seperti sebuah object.

Contoh Code Queue

- Header

```
#include <iostream>
#define MAX 20 //maksimum data queue
using namespace std;
```

- Deklarasi struct antrian

```
struct Queue {
   int front, rear, data[MAX];
}Q;
```

Memeriksa antrian

- cek apakah antrian penuh

```
bool isFull() {
   return Q.rear == MAX;
}
```

- cek apakah antrian kosong

```
bool isEmpty() {
   return Q.rear == 0;
}
```

CONT....

- Pada fungsi **`isFull()`** akan mengembalikan nilai true jika nilai Q.rear sama dengan maksimum data array yang telah ditentukan MAX, atau false jika tidak sama.
- Pada fungsi **`isEmpty()`** akan mengembalikan nilai true jika nilai Q.rear sama dengan 0, atau false jika tidak sama.

Menampilkan Antrian

- Menampilkan Queue

```
void printQueue() {
    if (isEmpty()) {
     cout << "Antrian kosong"<<endl;
    }
    else {
        cout << "QUEUE : ";
        for (int i = Q.front; i < Q.rear; i++)
        //menambahkan koma jika data tidak terdapat di
antrian pertama
            cout << Q.data[i] << ((Q.rear-1 == i) ? "" :
",");
        cout << endl;
  }
}
```

CONT....

Untuk menampilkan antrian, perlu diperiksa terlebih dahulu apakah antriannya kosong. Jika kosong maka tidak ada data untuk ditampilkan, jadi cukup tampilkan pesan. Tapi jika antrian berisi data atau ada antrian disana maka tampilkan data yang ada di antrian menggunakan **for** loop.

Input Data ke antrian

- Menambahkan data ke antrian

```
void enqueue() {
    if (isFull())
    {
        cout << "Antrian penuh!"<<endl;
    }
    else {
        int data;
        //menambahkan data ke antrian
        cout << "Masukkan Data : ";cin >> data;
        Q.data[Q.rear] = data;
        //menempatkan tail pada elemen data terakhir yang
ditambahkan
        Q.rear++;
        cout << "Data ditambahkan\n";
        printQueue();
    }
}
```

CONT....

- Dalam menginputkan antrian hal pertama yang harus di lakukan adalah memriksa apakah antrean full dengan `IsFull()`
- Jika masih ada ruang maka inputkan data ke antrian dan tambahkan satu nilai ke `Q.rear` dimana data tersebut berada pada antrian paling belakang.
- `printQueue()` berfungsi untuk menampilkan data antrean yang telah diinputkan

Mengambil Data antrian

- mengambil data dari antrian

```
void dequeue() {
    if (isEmpty())
    {
        cout << "Antrian masih kosong"<<endl;
    }
    else{
        cout << "Mengambil data \"" << Q.data[Q.front] <<
"\"..." << endl;
        //menggeser antrian data ke head
        for (int i = Q.front; i < Q.rear; i++)
            Q.data[i] = Q.data[i + 1];
        //menempatkan tail pada data terakhir yang digeser
        Q.rear--;
        printQueue();
    }
}
```

- Dalam menghapus/mengambil data antrian, yang pertama harus dilakukan adalah melakukan pengecekan apakah data berisi atau kosong dengan `isEmpty()`
- Jika ada data pada antrian, maka geser data ke antrian palng depan atau `Q.front,`
- Data yang keluar dari antrian paling depan akan ditimpa dan nila `Q.rear` dikurangi satu nilai

Menampilkan Menu

```
int main() {
   int choose;
   do
   {
      //Tampilan menu
      cout << "--------------------\n"
         << "   Menu Pilihan\n"
         << "--------------------\n"
         << " [1] Enqueue \n"
         << " [2] Dequeue\n"
         << " [3] Keluar \n\n"
         << "--------------------\n"
         << "Masukkan pilihan : "; cin >> choose;
```

```
switch (choose)
      {
      case 1:
         enqueue();
         break;
      case 2:
         dequeue();
         break;
      default:
         cout << "Pilihan tidak tersedia";
         break;
      }
   } while (choose !=3);
   return 0;
}
```