

# Queue

## Tujuan Praktikum

1. Memahami Queue dalam program,
2. Mampu menerapkan Queue untuk menyelesaikan berbagai kasus.

## Dasar Teori

Queue bersifat FIFO (First In First Out) yaitu elemen pertama yang ditempatkan pada queue adalah yang pertama dipindahkan.

Operasi-operasi antrian

- CREATE  
Untuk menciptakan dan menginisialisasi queue dengan cara membuat Head dan Tail = -1
- ISEMPY  
Untuk memeriksa apakah queue kosong
- ISFULL  
Untuk memeriksa apakah queue sudah penuh
- ENQUEUE  
Untuk menambahkan item pada posisi paling belakang
- DEQUEUE  
Untuk menghapus item dari posisi paling depan
- CLEAR  
Untuk mengosongkan queue

## Praktik (Guided)

1. Compile program dibawah ini !

2. Berikan penjelasan pada masing-masing fungsi yang terdapat pada program.  
Jelaskan apa kegunaan masing-masing fungsi !

```
#include <stdio.h>
#define MAX 8

typedef struct {
    int data[MAX];
    int head;
    int tail;
} Queue;

Queue antrian;

void Create() {
    antrian.head = antrian.tail = -1;
}

int IsEmpty() {
    if(antrian.tail == -1)
        return 1;
    else
        return 0;
}

int IsFull() {
    if(antrian.tail == MAX-1)
        return 1;
    else
        return 0;
}

void Enqueue(int data) {
    if(IsEmpty() == 1) {
        antrian.head = antrian.tail = 0;
        antrian.data[antrian.tail] = data;
        printf("%d masuk!\n", antrian.data[antrian.tail]);
    } else {
        if(IsFull() == 0) {
            antrian.tail++;
            antrian.data[antrian.tail] = data;
            printf("%d masuk!\n", antrian.data[antrian.tail]);
        }
    }
}

int Dequeue() {
    int i;
    int e = antrian.data[antrian.head];
    for(i = antrian.head; i <= antrian.tail - 1; i++) {
        antrian.data[i] = antrian.data[i+1];
    }
}
```

```

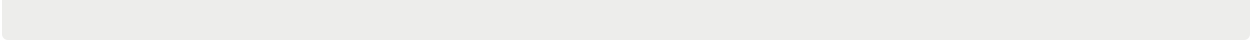
    }
    antrian.tail--;
    return e;
}

void Clear() {
    antrian.head=antrian.tail=-1;
    printf("data clear\n");
}

void Tampil() {
    if(IsEmpty()==0) {
        int jum = 0;
        for(int i=antrian.head;i<=antrian.tail;i++) {
            printf("%d ",antrian.data[i]);
            jum=jum+antrian.data[i];
        }
        printf("\n");
    } else {
        printf("data kosong!\n");
    }
}

int main() {
    int pil;
    int data;
    Create();
    do {
        printf("\n");
        printf("1. Enqueue\n");
        printf("2. Dequeue\n");
        printf("3. Tampil\n");
        printf("4. Clear\n");
        printf("5. Exit\n");
        printf("Pilihan = ");
        scanf("%d",&pil);
        switch(pil) {
            case 1:
                printf("Data = ");
                scanf("%d",&data);
                Enqueue(data);
                break;
            case 2:
                printf("Elemen yang keluar : %d\n", Dequeue());
                break;
            case 3:
                Tampil();
                break;
            case 4:
                Clear();
                break;
        }
    } while(pil!=5);
}

```



## Latihan (Unguided)

3. Tambahkan fasilitas untuk menghitung banyaknya data, jumlah data, dan rata-rata dari keseluruhan data yang masuk ke dalam antrian !