

# Algoritma dan Struktur Data

## Queue

Umi Sa'adah

Tita Karlita

Entin Martiana K

Arna Fariza

2021



Politeknik Elektronika Negeri Surabaya  
Departemen Teknik Informatika dan Komputer

# Materi

- Definisi Queue
- Operasi pada Queue
- Representasi Queue

# Apakah Queue ?

- Abstract Data Type yang menerapkan konsep antrian.
- Merupakan konsep First In First Out (FIFO).
- Data yang disimpan pertama akan diambil lebih dahulu.




designed by  freepik

Image source <https://kreatip.id/materi/struktur-data/queue>

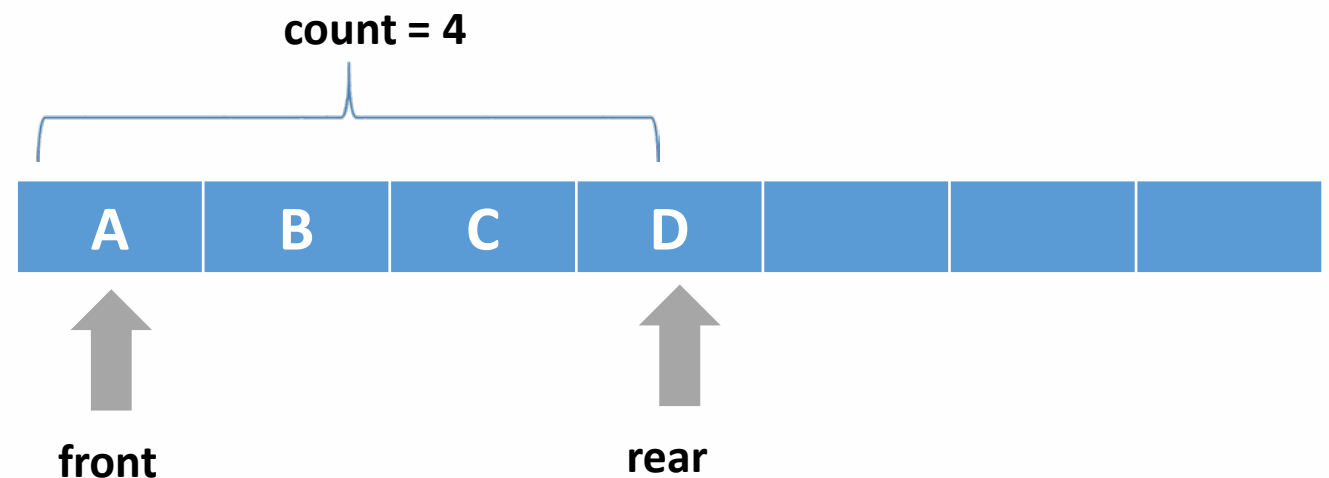
# Karakteristik Queue

- **Elemen** antrian yaitu data yang terdapat di elemen antrian.
- Elemen queue: A, B, C, dan D.



# Karakteristik Queue

- **Front** : penunjuk elemen terdepan dari antrian.
- **Rear** : penunjuk elemen terakhir dari antrian.
- **Count** : jumlah elemen pada antrian.



# Karakteristik Queue

Kondisi : penuh atau kosong

Kondisi Penuh



count = MAX

Kondisi Kosong



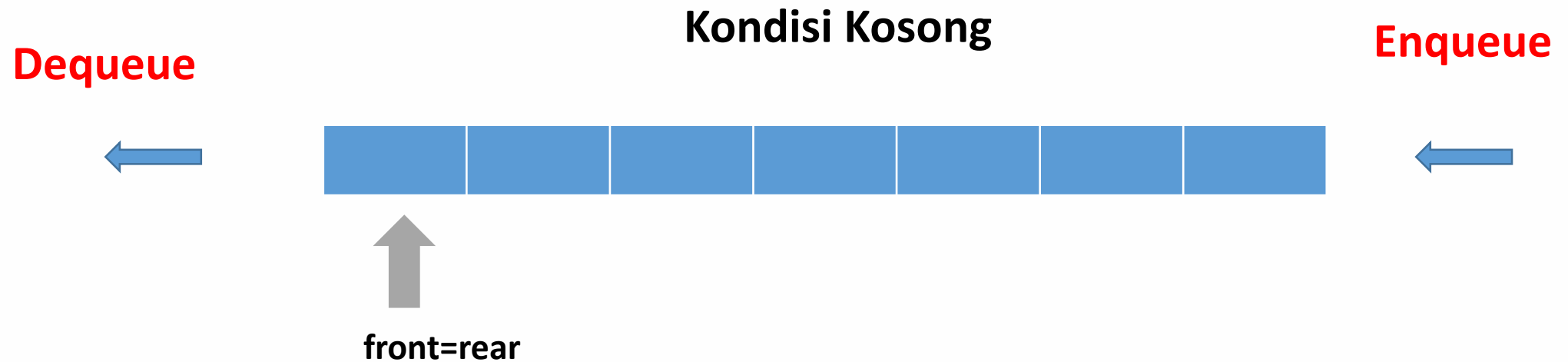
count = 0

# Representasi Queue

- Menggunakan:
  - array
  - linked list
- Pada implementasi queue dengan array, kemungkinan queue bisa penuh
- Pada implementasi queue dengan linked list, queue tidak pernah penuh

# Operasi Queue

- **enqueue**: memasukkan data ke antrian
- **dequeue**: mengeluarkan data dari antrian





# Operasi Queue



# Operasi Queue



**Enqueue(B)**



# Operasi Queue



Enqueue(C)



# Operasi Dequeue

Dequeue()

A



# Operasi Dequeue

Dequeue()

B



# Operasi Dequeue

Dequeue()

C



# Representasi Queue dengan Array

# Representasi Queue dengan Array

```
#define MAX 7
typedef char itemType
typedef struct {
    itemType item[MAX];
    int count;
    int front;
    int rear;
} Queue;
```



# Operasi pada Queue

- enqueue : menyimpan item ke Queue
- dequeue : menghapus item dari Queue
- inisialisasi : inisialisasi awal Queue
- penuh : mengecek apakah queue dalam kondisi penuh
- kosong : mengecek apakah queue dalam kondisi kosong

# Operasi Inisialisasi

## Menginisialisasi

- **count** sama dengan 0.
- **front** menunjuk ke indeks 0.
- **rear** menunjuk ke indeks 0.

```
void inisialisasi (Queue *q) {
    q->count=0;
    q->front = 0;
    q->rear = 0;
}
```

**count = 0**

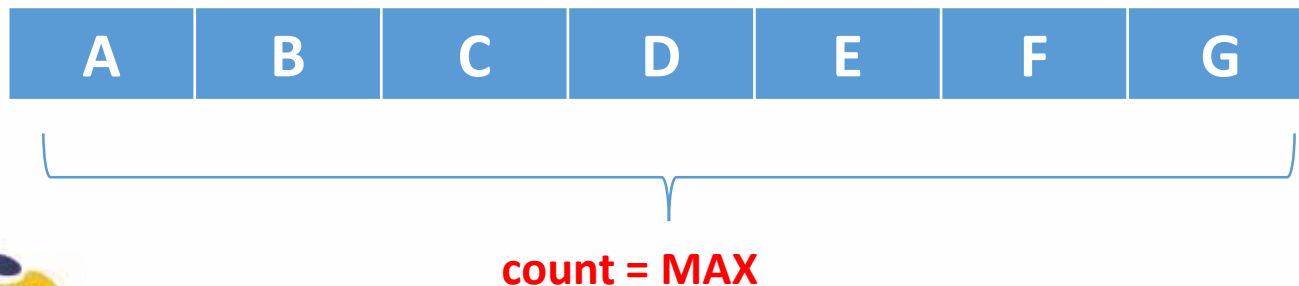


**front=rear=0**

# Operasi Penuh

- Melakukan pengecekan apakah Queue
  - penuh (**count** bernilai = **MAX**), atau
  - tidak penuh (**count** bernilai < **MAX**),
- Jika penuh return value=1, sebaliknya return value=0
- Digunakan saat melakukan operasi ENQUEUE

Kondisi Penuh



```
int penuh (Queue *q) {  
    if (q->count==MAX)  
        return 1;  
    else  
        return 0;  
}
```

# Operasi Kosong

- Melakukan pengecekan apakah queue
  - Kosong (**count** bernilai **= 0**),
  - Tidak kosong (Jika kosong return value=1, sebaliknya return value=0)
- Digunakan bila melakukan operasi DEQUEUE

Kondisi Kosong

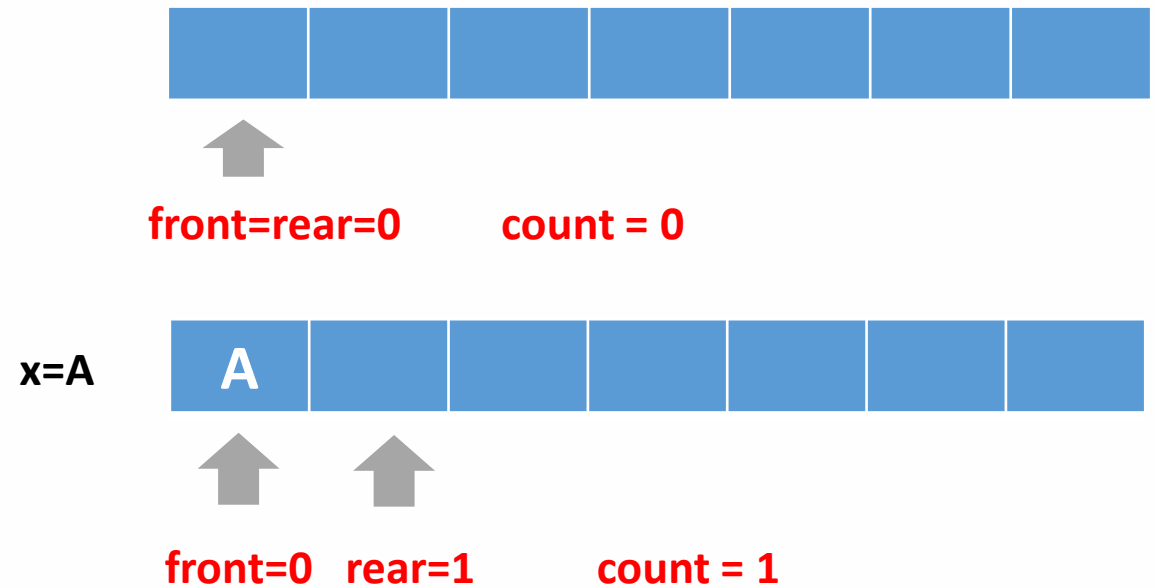


**count = 0**

```
int kosong (Queue *q) {
    if (q->count==0) ;
        return 1;
    else
        return 0;
}
```

# Operasi ENQUEUE

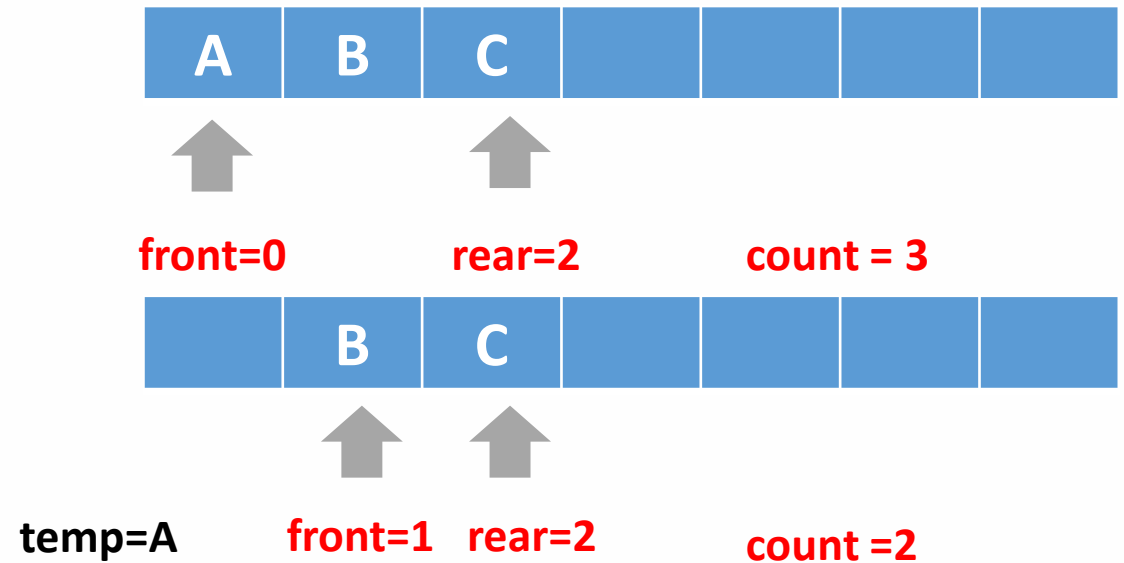
- Jika array penuh (**count=MAX**), tidak dapat melakukan operasi Enqueue.
- Menyimpan data pada posisi **rear**.
- Setelah dilakukan penyimpanan, posisi rear di-increment. (**rear++**).
- Circular queue  $\rightarrow$  (**rear++**) % MAX.
- Jumlah elemen diincrement (**count++**).



```
void Enqueue (Queue *q, itemType x){
    if(penuh(q))
        printf("Queue Penuh, data tidak dapat disimpan\n");
    else {
        q->item[q->rear]=x;
        q->rear = (q->rear+1) % MAX
        q->count++;
    }
}
```

# Operasi DEQUEUE

- Jika array **Kosong** (**count=0**), tidak dapat dilakukan operasi Dequeue.
- Mengambil data pada posisi **front**.
- Setelah mengambil data posisi **front** di-increment (**front++**)
- Circular queue  $\rightarrow$  (**front++**) % MAX.
- Jumlah elemen di-decrement (**count--**).



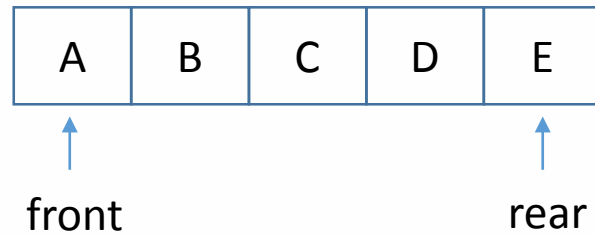
```

itemType Dequeue(Queue *q){
    itemType temp;
    if(Kosong(q)) {
        printf("Queue Kosong, tidak dapat mengambil data\n");
        return ' ';
    }else {
        temp=q->item[q->front];
        q->front = (q->front+1) % MAX;
        q->count--;
        return(temp);
    }
}

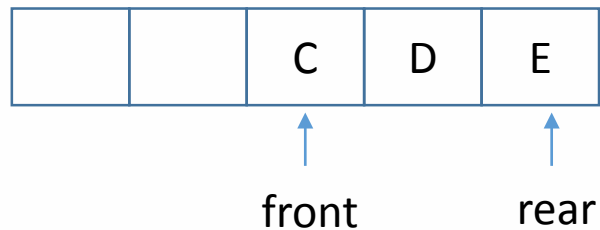
```

# Circular Queue

Enqueue(A)  
Enqueue(B)  
Enqueue(C)  
Enqueue(D)  
Enqueue(E)

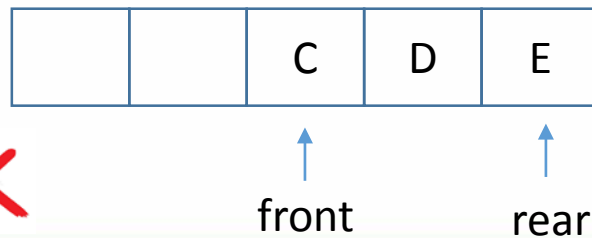


Dequeue()  
Dequeue()

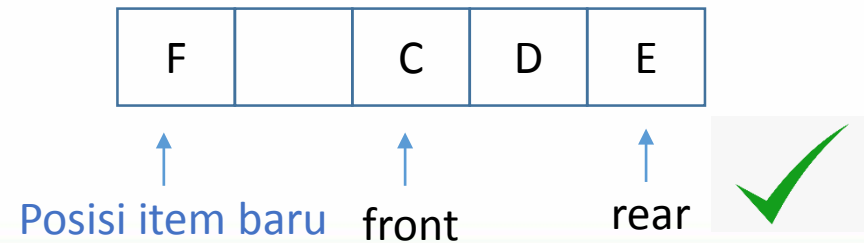


## Linier Queue

Enqueue(F)



## Circular Queue





# Circular Array-based Queue - Example

**ENQUEUE**  
`q->item[q->rear]=x;`  
`q->rear = (q->rear+1) % MAX;`  
`q->count++;`

● front  
● rear

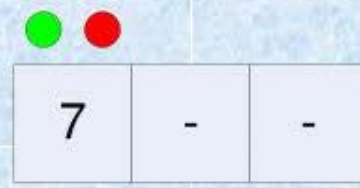
- Circular array with NMAX=3 entries

front=rear=0  
count=0



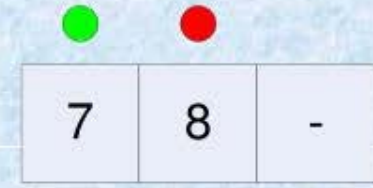
enqueue(7)

item[0]=7  
front=0, rear=0+1%3=1  
count=1



enqueue(8)

item[1]=8  
front=0, rear=1+1%3=2  
count=2



item[2]=6  
front=0, rear=2+1%3=0  
count=3

enqueue(6)



temp=item[0]→7  
front=0+1%3=1, rear=0  
count=2

dequeue()  
returns 7



peek()  
returns 8

temp=item[1]→8  
front=1+1%3=2, rear=0  
count=1

dequeue()  
returns 8



item[0]=4  
front=2, rear=0+1%3=1  
count=2

enqueue(4)



enqueue(10)

item[1]=10  
front=2, rear=1+1%3=2  
count=3



dequeue()  
returns 6

temp=item[2]→6  
front=2+1%3=0, rear=2  
count=2



enqueue(13)

item[2]=13  
front=0, rear=2+1%3=0  
count=3



**DEQUEUE**  
`temp=q->item[q->front];`  
`q->front = (q->front+1)%MAX;`  
`q->count--;`  
`return(temp);`



# Rangkuman

- Queue merupakan konsep penyimpanan elemen secara FIFO.
- Elemen yang masuk lebih awal akan keluar lebih dahulu
- Komponen pada Queue terdiri dari :
  - Elemen yang disimpan di penyimpanan
  - penunjuk depan (front)
  - penunjuk belakang (rear)
  - jumlah item (count)
- Operasi pada Queue : ENQUEUE dan DEQUEUE
- Operasi tambahan pada Queue : Inisialisasi, Penuh, Kosong
- Representasi queue:
  - Array
  - Linked list



# Referensi

1. Brian W. Kernighan, Dennis M. Ritchie (2012): The C Programming Language : Ansi C Version 2 Edition, PHI Learning
2. Byron Gottfried (2010) : Programming with C, Tata McGraw - Hill Education
3. [Kochan Stephen](#) (2004) : Programming in C, 3rd Edition, Sams
4. K. N. King (2008) : C Programming: A Modern Approach, 2nd Edition, W. W. Norton & Company
5. Abdul Kadir (2012) : Algoritma & Pemrograman Menggunakan C & C++, Andi Publisher, Yogyakarta
6. <http://www.gdsw.at/languages/c/programming-bbrow/>
7. <https://www.petanikode.com/tutorial/c/>
8. <http://www.cprogramming.com/tutorial/c-tutorial.html>



**bridge to the future**

<http://www.eepis-its.edu>