

# Pertemuan 9

## QUEUE (ANTREAN)



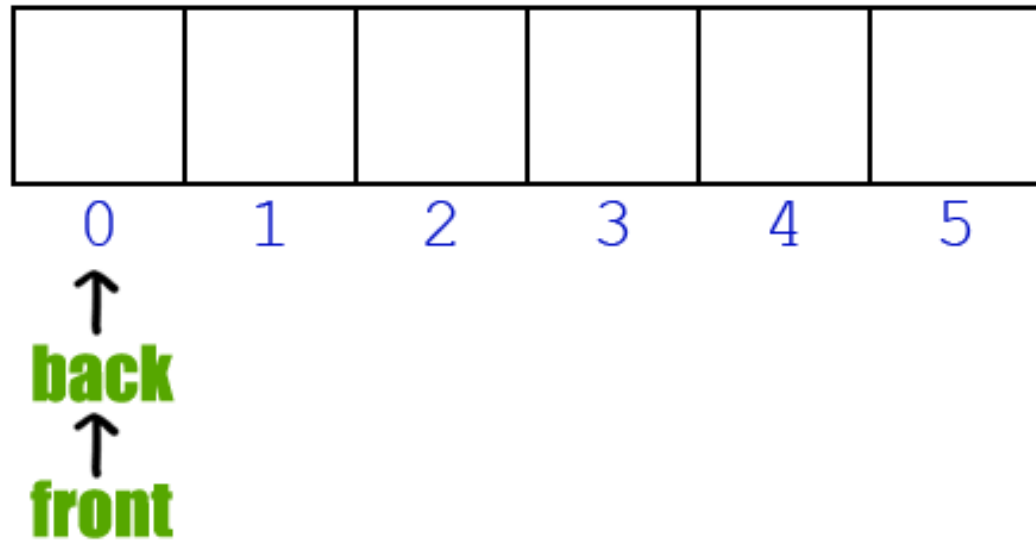
## PENGERTIAN QUEUE (ANTREAN)

Struktur Data Antrean (Queue) adalah suatu bentuk khusus dari List Linier dengan operasi pemasukan data hanya diperbolehkan pada salah satu sisi, yang disebut sisi Belakang / ekor (Tail) dan operasi penghapusan hanya diperbolehkan pada sisi lainnya yang disebut sisi Depan / kepala (Head) dari LinkedList.

Prinsip Antrean : FIFO (First In First Out)  
FCFS (First Come First Serve)

***“Yang Tiba lebih awal Maka akan dilayani Terlebih Dahulu”***

# Deklarasi Queue



# Queue dengan Python

```
1 from collections import deque # import module
2 antrian = deque([1,2,3,4,5])
3 print('data antrian sekarang : ', antrian)
4
5 #menambahkan data antrian
6 antrian.append(6)
7 print('data antrian masuk : 6')
8 print('data antrian : ', antrian)
9
10 #mengurangi antrian
11 keluar = antrian.popleft()
12 print('data keluar : ', keluar)
13 print('data antrian sekarang : ', antrian)
```

## Hasil



data antrian sekarang : deque([1, 2, 3, 4, 5])



data antrian masuk : 6



data antrian : deque([1, 2, 3, 4, 5, 6])

data keluar : 1



data antrian sekarang : deque([2, 3, 4, 5, 6])

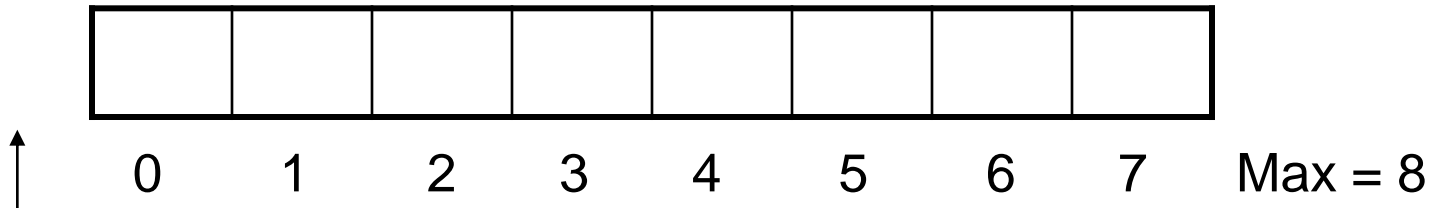
# OPERASI QUEUE

- **PEEK ()**  
Digunakan untuk mendapatkan elemen yang terdapat pada posisi terakhir tanpa menghapus element tersebut.
- **ISEMPTY()**  
Untuk memeriksa apakah queue kosong
- **SIZE()**  
mengembalikan jumlah item di dalam list. Tidak memerlukan parameter dan mengembalikan suatu integer.
- **ENQUEUE()**  
Untuk menambahkan item pada posisi paling belakang
- **DEQUEUE()**  
Untuk menghapus item dari posisi paling depan

# Inisialisasi Head

Digunakan untuk membentuk dan menunjukan awal terbentuknya suatu Antrean / Queue

```
def __init__(self):  
    self.head = Node("head")  
    self.size = 0
```



Antrian pertama kali

# Fungsi IsEmpty

- Untuk memeriksa apakah Antrian penuh atau kosong
- Head adalah tanda untuk kepala antrian (elemen pertama dalam antrian) yang tidak akan berubah-ubah
- Pergerakan pada Antrian terjadi dengan penambahan elemen Antrian kebelakang,

# Fungsi isEmpty (Lanjutan)

```
def isEmpty(self):  
    return self.size == 0
```



0

1

2

3

4

5

6

7

Max = 8

Antrian kosong



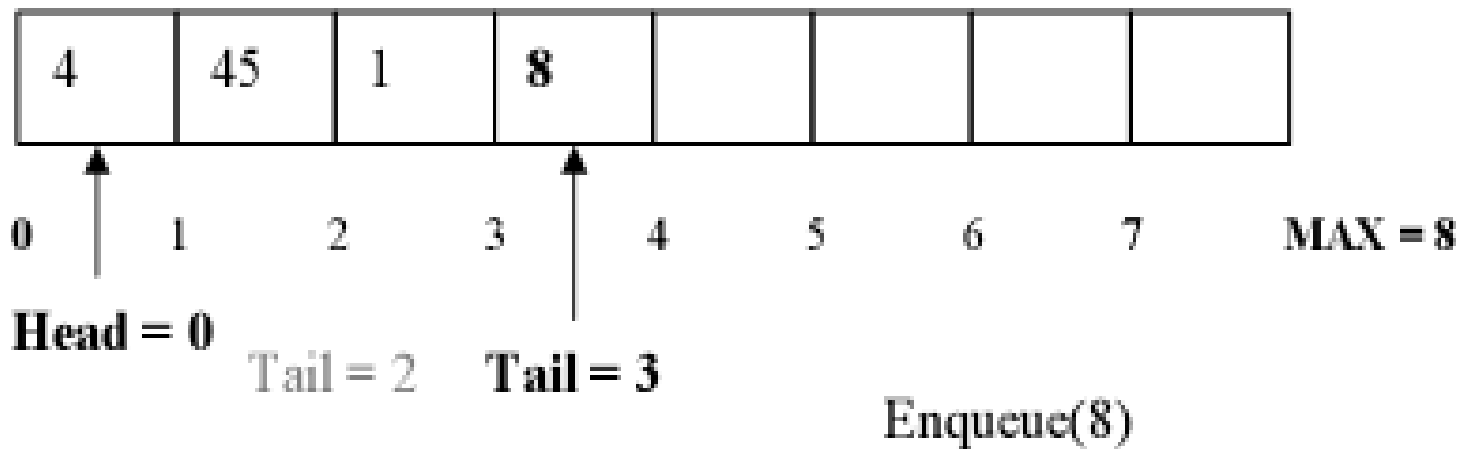
# Fungsi Peek

Periksa antrian untuk melihat apakah ada antrian kosong

```
def peek(self):  
  
    if self.isEmpty():  
        raise Exception("Peeking from an empty stack")  
    return self.head.next.value
```

# Fungsi Enqueue

- Untuk menambahkan elemen ke dalam Antrian, penambahan elemen selalu dilakukan pada elemen paling **belakang**
- Penambahan elemen selalu menggerakkan variabel Tail dengan cara menambahkan Tail terlebih dahulu



# Fungsi Enqueue (Lanjutan)

Menambahkan nilai pada antrian

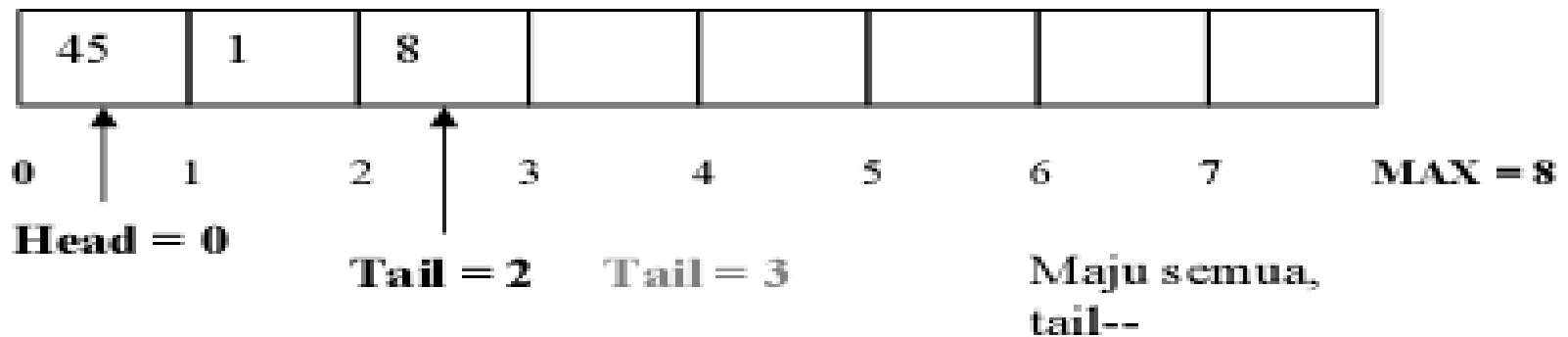
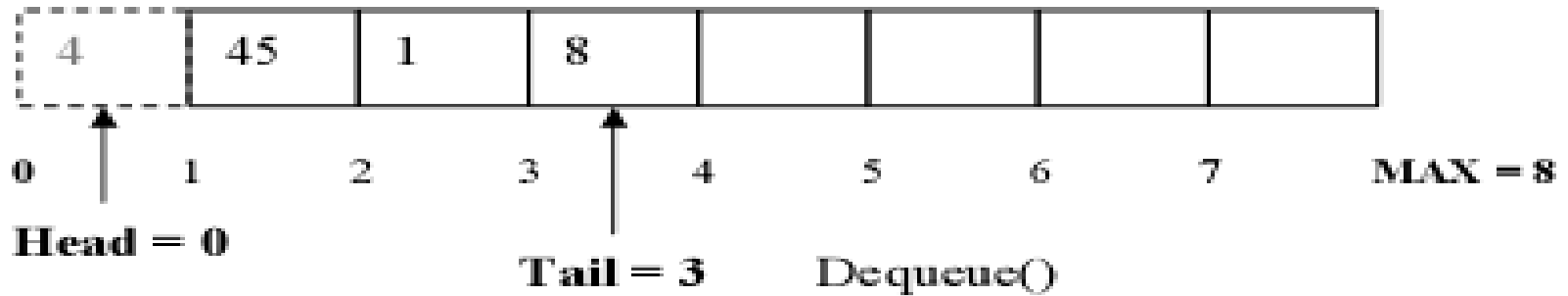
```
def enqueue(self, value):  
    node = Node(value)  
    node.next = self.head.next  
    self.head.next = node  
    self.size += 1
```

# Fungsi Dequeue

- Digunakan untuk menghapus elemen terdepan (head) dari Antrian
- Dengan cara : menggeser semua elemen antrian kedepan dan mengurangi Tail dgn 1. Penggeseran dilakukan dengan menggunakan looping

```
def dequeue(self):  
    if self.isEmpty():  
        raise Exception("dequeue from an empty stack")  
    remove = self.head.next  
    self.head.next = self.head.next.next  
    self.size -= 1  
    return remove.value
```

# Fungsi Dequeue (Lanjutan)



# Fungsi SIZE

digunakan untuk mengetahui banyak elemen atau data yang ada di dalam list.

```
def getSize(self):  
    return self.size
```

## Latihan

Berikan gambaran/ilustrasi dari kasus antrian berikut :

- a) Diketahui suatu Antrian/queue dgn  $\text{max} = 6$ .
- b) Lakukan Enqueue 4 elemen ke dalam antrian, dimanakah posisi Head dan Tail ?
- c) Kemudian lakukan Dequeue 2 elemen dari antrian. Maka dimana posisi Head dan Tail ?
- d) Dari keadaan diatas, bagaimanakah kondisi penuh dan IsEmpty nya ?