```
Type TQueue = <wadah:array[1..10] of characte Procedure Enqueue(input/output Q:TQueue, input
         head:integer,
         tail:integer >
{Queue model I, kondisi head 0 atau 1}
Procedure CreateQueue(output Q:TQueue)
{I.S: -; F.S: Q terdefinisi}
{Proses: mengisi elemen wadah dengan '',
 head 0, tail 0}
Function Head(Q:TQueue) -> integer
{mengembalikan posisi elemen terdepan}
Function Tail(Q:TQueue) -> integer
{mengembalikan posisi elemen terakhir}
Function InfoHead(Q:TQueue) -> character
{mengembalikan nilai elemen terdepan}
Function InfoTail(Q:TQueue) -> character
{mengembalikan nilai elemen terakhir}
Function isEmptyQueue(Q:TQueue) -> boolean
{mengembalikan true bila Q kosong}
Function isFullQueue(Q:TQueue) -> boolean
{mengembalikan true bila Q penuh}
```

```
e:character)
                                              {I.S: Q,e terdefinisi, Q mungkin kosong }
                                             {F.S: Q tetap, atau infoTail(Q)=e }
{pergeseran maju pada elemen ketika dequeue} {Proses menambah elemen e ke ekor Q bila belum
                                              penuh}
                                              Procedure Dequeue(input/output Q:TQueue, output
                                              e:character)
                                              {I.S: Q terdefinisi, mungkin kosong }
                                              {F.S: Q tetap, atau e berisi infoHead(Q) lama }
                                             Proses menghapus elemen e dari head Q bila belum
                                              kosong}
                                              {lalu geser maju 1 langkah semua elemen di belakang
                                              head}
                                              Procedure PrintQueue(input Q:TQueue)
                                              {I.S:-; F.S:-; Proses: menampilkan kondisi wadah Q }
                                              Procedure ViewQueue(input Q:TQueue)
                                              {I.S:-; F.S:-; Proses: menampilkan info elemen tak
                                              kosong Q}
                                              Function sizeQueue(Q:TQueue) -> integer
                                              {mengembalikan panjang/banyak elemen}
```

## Procedure CreateQueue(output Q:TQueue)

```
{I.S: -; F.S: Q terdefinisi}
{Proses: mengisi elemen wadah dengan '',head 0, tail 0}
```

## Kamus Lokal

i : integer

## Algoritma

Q.head <-- 0 Q.tail <-- 0

i traversal [1..10] Q.wadah[i] <-- ' ' Function Head(Q:TQueue) -> integer {mengembalikan posisi elemen terdepan}

Kamus lokal { - }

Algoritma

--> Toueue.head

Kamus lokal

Algoritma

-> Q.head

```
Function Tail(Q:TQueue) -> integer {mengembalikan posisi elemen terakhir} 
Kamus lokal
```

Algoritma
--> Q.tail

{ - }

```
Function InfoHead(Q:TQueue) -> character {mengembalikan nilai elemen terdepan}

Kamus lokal { - }
```

## Algoritma if NOT isEmpty(Q) then --> Q.wadah[Q.head] else --> ''

```
Function InfoTail(Q:TQueue) -> character 
{mengembalikan nilai elemen terakhir}
```

```
Kamus lokal
{ - }

Algoritma

if NOT isEmpty(Q) then

--> Q.wadah[Q.tail]

else
--> '''
```

```
Function isEmptyQueue(Q:TQueue) > boolean 
{mengembalikan true bila Q kosong}
```

```
Kamus Lokal
   empty: boolean
   i : integer
Algoritma
   empty <-- true
   i traversal [1..10]
    if (Q.wadah[i]!='') then
       empty <-- false
   --> empty
```

```
-> Head(Q) = 0 AND Tail(Q) = 0
```

```
empty <-- true
i = 1
while ( empty = true AND i <= 10 ) do
    if ( Q.wadah[i] != ' ' )
        empty <-- false
    i = i + 1
--> empty
```

```
Function isFullQueue(Q:TQueue) -> boolean
  {mengembalikan true bila Q penuh}
Kamus Lokal
   full: boolean
   i : integer
Algoritma
   full <-- true
   i traversal [1..10]
    if ( Q.wadah[i] = ' ' ) then
       full <-- false
```

```
\rightarrow Head(Q) = 1 AND Tail(Q) = 10
```

--> full

```
Procedure Enqueue(input/output Q:TQueue, input e:character)
{I.S: Q,e terdefinisi, Q mungkin kosong }
                                                                      cek 4 kondisi:
{F.S: Q tetap, atau infoTail(Q)=e }
                                                                    - penuh
{Proses menambah elemen e ke ekor Q bila belum penuh}
                                                                     - kosong √
if not isFullQueue(Q) then
  if isEmptyQueue(Q) then
```

head (Q)  $\leftarrow$  1 Head(Q)  $\leftarrow$  Head(Q) + 1

else

 $tail(Q) \leftarrow tail(Q)+1$ 

Q.wadah[tail(Q)]<--e

Procedure Dequeue(input/output Q:TQueue, output e:character)

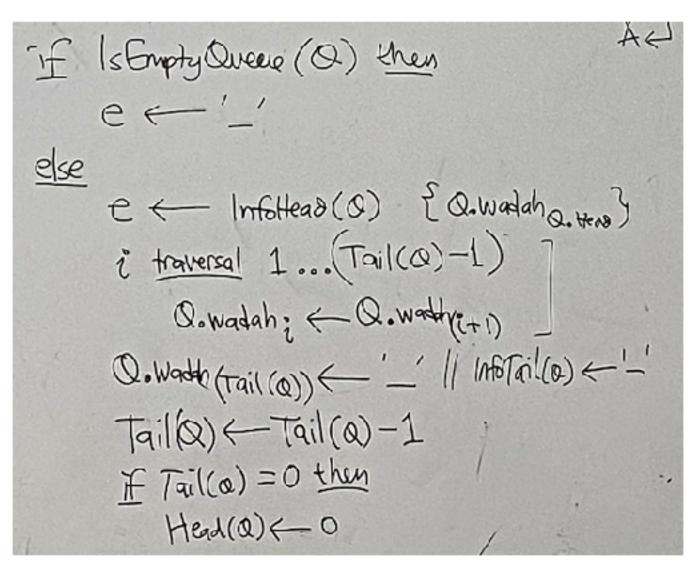
{I.S: Q terdefinisi, mungkin kosong }

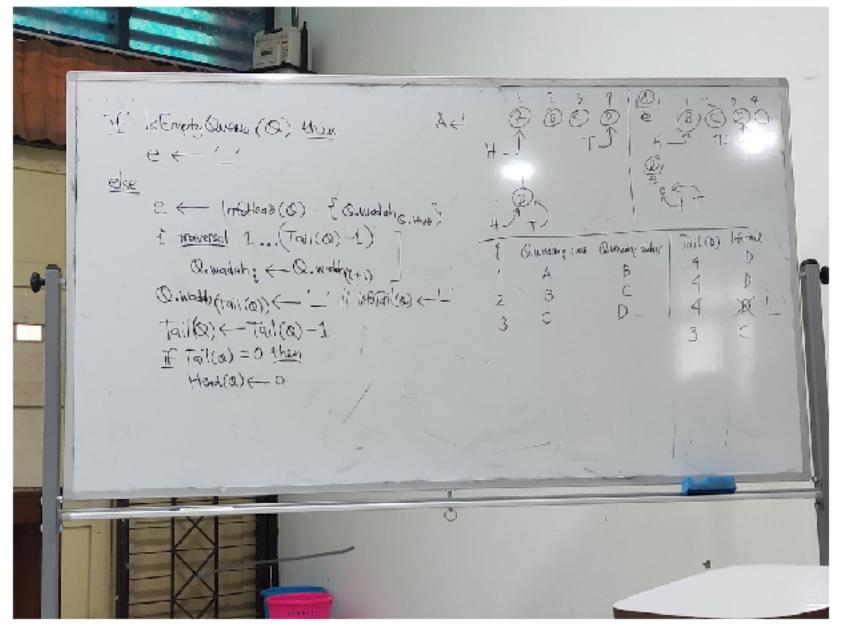
{F.S: Q tetap, atau e berisi infoHead(Q) lama }

(Proses menghapus elemen e dari head Q bila belum kosong)

{lalu geser maju 1 langkah semua elemen di belakang

head}





```
Procedure PrintQueue(input Q:TQueue)
{I.S:-; F.S:-; Proses: menampilkan kondisi wadah Q }

Kamus Lokal
i: integer

Algoritma
i traversal [1..10]
output (Q.wadah[i])
```

```
Procedure ViewQueue(input Q:TQueue)
{I.S:-; F.S:-; Proses: menampilkan info elemen tak
kosong Q}
Kamus Lokal
i : integer
Algoritma
if (Q.head = 1) then
  i traversal [1..Q.tail]
     output (Q.wadah[i])
 else
  output "antrean kosong"
```

Function sizeQueue(Q:TQueue) -> integer {mengembalikan panjang/banyak elemen}

```
Kamus lokal
Finteger
Pinteger
                                      Kamus lokal
                                      i : integer
Algorithma

-> Q. fail
                                      P: integer
                                      Algoritma
                                      P <- 0
                                      if(Q.head = 1) then
                                        i trave<del>rs</del>al [1 .. T.tail]
                                            if(Q.wadah[i] =/= ' ')
                                              P = P
                                        ->P
```