```
Type TQueue = <wadah:array[1..10] of character,
              head:integer,
              tail:integer >
{Queue model I, kondisi head 0 atau 1}
{pergeseran maju pada elemen ketika dequeue}
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}
```

```
Procedure Enqueue(input/output Q:TQueue, input
e:character)
{I.S: Q,e terdefinisi, Q mungkin kosong }
{F.S: Q tetap, atau infoTail(Q)=e }
{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 {iterator}
algoritma
    i traversal 1..10
        Q.wadah[i] <-- ‘ ’
    Q.head <-- 0
    Q.tail <-- 0
```

```
Function Head(Q:TQueue) -> integer
{mengembalikan posisi elemen terdepan}

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 (Q.head = 1) then
    -> Q.wadah[1]
  else
    -> ‘ ’

Q.wadah[Q.head]

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

kamus lokal

algoritma
    if (NOT(isEmptyQueue(Q))) then
        -> Q.wadah[Q.tail]
    else
        -> ‘ ’
```

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

kamus lokal

algoritma
    if (Q.head = 0) then     AND Q.tail=0
        -> true
    else
        -> false
```

```
Function isFullQueue(Q:TQueue) -> boolean
{mengembalikan true bila Q penuh}

kamus lokal

algoritma
   if (Q.tail = 10) then    AND Q.head=1
      -> true
   else
     -> false
```

Procedure Enqueue(input/output Q:TQueue, input e:character)
{I.S: Q,e terdefinisi, Q mungkin kosong }
{F.S: Q tetap, atau infoTail(Q)=e }
{Proses menambah elemen e ke ekor Q bila belum penuh}

kamus lokal

algoritma

```
if (NOT(isFullQueue(Q))) then
    Q.tail <-- Q.tail + 1
    Q.wadah[Q.tail] <-- e
    if Q.head=0 then
        Q.head <-- 1
else
    output "Queue tetap"
```

```
if NOT isFullQueue(Q) then
    if isEmptyQueue(Q) then
        Q.head <-- 1
    Q.tail <-- Q.tail + 1
    Q.wadah[Q.tail] <-- 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}

kamus lokal
    i, j: integer
algoritma
if(NOT(isEmptyQueue(Q))) then
    e <-- Q.wadah[Q.head]
    i traversal 1..Q.tail-1          { geser elemen }
        Q.wadah[i] <-- Q.wadah[i+1]
    Q.wadah[Q.tail] <-- ' '
    Q.tail <-- Q.tail -1
    if Q.tail = 0 then {kasus 1 elemen, maka Q menjadi kosong}
        Q.head <-- 0
else
    output "Queue kosong'
    e <-- ' '
```

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


  kamus Lokal
  i : integer

  algoritma
  i traversal 1..10
       -> 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 NOT isEmptyQueue(Q)     then
      i traversal 1..Q.tail
         if Q.wadah[i] =/= ' '  then
            output Q.wadah[i]
      -> Q.wadah[i]
else
      output "queue kosong"
```

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

kamus Lokal
    i : integer
    j : integer

algoritma
    -> Q.tail