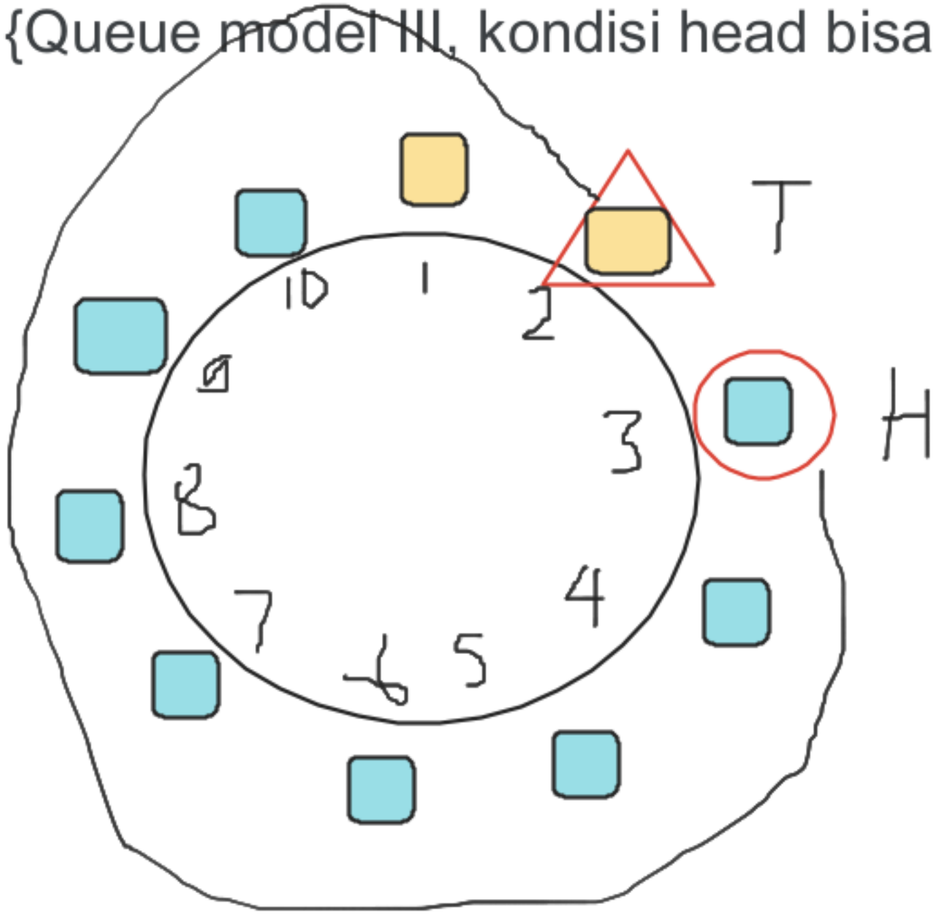
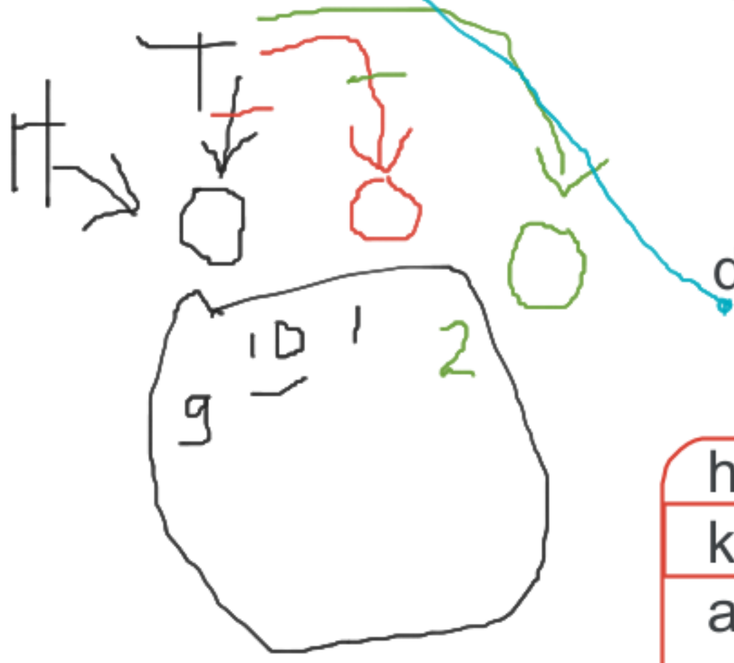
Type TQueue3 = <wadah:array[1..10] of character, head:integer, tail:integer >
{Queue model III, kondisi head bisa berputar}



isfull:
H 1 T 10
H 3 T 2
H 10 T 9

enqueue:
10 mod 10 + 1
2 mod 10 + 1
9 mod 10 + 1



dequeue:
10 mod 10 + 1

- kasus kosong
- kasus tidak kosong
  - kasus penuh
  - kasus tidak penuh
  - kasus 1 elemen

header
kamus
algoritma

```
Function InfoHead(Q:Tqueue3) -> character
if isEmptyQueue(Q) then
    -> '@'
    else
     -> Q.wadah[Q.head]
{mengembalikan nilai elemen terdepan}
Function InfoTail(Q:Tqueue3) -> character
if isEmptyQueue(Q) then
    -> '@'
    else
     -> Q.wadah[Q.tail]
{mengembalikan nilai elemen terakhir}
Function isOneElement(Q:Tqueue3) -> boolean
if head(Q) = 1 AND tail(Q) = " "
   -> True
{mengembalikan true bila Q 1 elemen}
```

```
Function InfoHead(Q:Tqueue3) -> character
{mengembalikan nilai elemen terdepan}

Kamus lokal

Algoritma
  if isEmptyQueue(Q) then
    --> '@'
  else
    --> Q.wadah[Q.head]
```

sonni

Header fungsi:
definisi
spesifikasi

kamus lokal

algoritma

```
Function InfoTail(Q:Tqueue3) -> character
{mengembalikan nilai elemen terakhir}

Kamus lokal

Algoritma
  if isEmptyQueue(Q) then
    --> '@'
  else
    --> Q.wadah[Q.tail]
```

Kamus Lokal

```
function isOneElement(Q:Tqueue3) -> boolean
{mengembalikan true bila Q 1 elemen}
kamus lokal
    ret: boolean
algoritma
    ret <-- false
    if NOT isEmptyQueue(Q) then
        ret <-- Q.tail = Q.head
    --> ret
```

```
Function isOneElement(Q:Tqueue3) -> boolean
{mengembalikan true bila Q 1 elemen}

Kamus Lokal
  ret : boolean

Algoritma
  ret <-- false
  if  NOT isEmptyQueue(Q) then
    ret <-- Q.tail = Q.head
  --> ret
```

50%

Function isFullQueue(Q:Tqueue3) -> boolean
{mengembalikan true bila Q penuh}

kamus lokal

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

Rafli
kamus lokal

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

vicky

if (ekspresi logika) then
- operator logika: AND, OR, NOT
- operator relasional: =, >, <, =/=, <=, >=

H 1 T 10
H 3 T 2
H 10 T 9

```
Procedure Enqueue(input/output Q:Tqueue3, 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}
{Bila Tail lama di kapasitas maka Tail baru menjadi 1}

kamus lokal



algoritma
    if NOT isFullQueue(Q) then
        if isEmptyQueue(Q) then { jika queue kosong maka set head dan tail menjadi 1 }
            Q.tail <-- 1
            Q.head <-- 1
        Q.tail <-- Q.tail mod 10 + 1 { operasi yang dilakukan baik queue keadaan kosong maupun terisi sebagian }
        Q.wadah[Q.tail] <-- e
```

```
Procedure Dequeue(input/output Q:Tqueue3, 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}
{bila Head di kapasitas, maka Head baru menjadi 1}

kamus lokal

algoritma
 if (isEmptyQueue(Q)) then {kosong}
   e <-- '@'
 else {tidak kosong}
   e <-- Q.wadah[Q.head]
   if (isOneElement(Q)) then {1 elemen}
     Q.head <-- 0
     Q.tail <-- 0
   else {banyak elemen}
     Q.head <-- Q.head mod 10 + 1
```

rafli

fungsi vs prosedur
parameter formal vs aktual

```
Function SizeQueue(Q:Tqueue3) --> integer
{mengembalikan ukuran antrian}

Kamus lokal
  N : integer
  H : integer
  T : integer

Algoritma
  H <-- Q.head
  T <-- Q.tail
  N <-- 0

  if isFullQueue(Q) then
    N <-- N + 10
  else
    i traversal 1 .. 10
      if Q.wadah[i] != '@' then
        N <-- N + 1

  --> N
```

berbasis konten

steven

```
aufarizq
kamus lokal
    size : integer

algoritma
    if isEmptyQueue(Q) then
        size <-- 0
    else
        if Q.head <= Q.tail then
            size <-- Q.tail - Q.head + 1
        else
            size <-- 10 - Q.head + Q.tail + 1
    --> size
```