

Type Tstack = <wadah:array[1..10] of character, top:integer>

Procedure CreateStack(output S:Tstack)

{I.S: - ; F.S: S terdefinisi }

{Proses mengisi elemen wadah dengan ' ', top 0}

Kamus lokal

i : integer

Algoritma

S.top <- 0

i traversal 1..10

S.wadah[i] <-- ' '

Type Tstack = <wadah:array[1..10] of character, top:integer>

Function Infotop(S:Tstack) -> character
{mengembalikan nilai elemen puncak}

Kamus Lokal

Algoritma

```
if (S.top /= 0) then  
    --> S.wadah[S.top]  
else { S.top = 0}  
    --> ''
```

Type Tstack = <wadah:array[1..10] of character, top:integer>

Function Top(S:Tstack) -> integer
{mengembalikan posisi puncak}

kamus lokal

algoritma
-->S.top

.

Type Tstack = <wadah:array[1..10] of character, top:integer>

Function isEmptyStack(S:Tstack) -> boolean
{mengembalikan true bila S kosong}

Kamus lokal

Algoritma

--> S.top = 0

Type Tstack = <wadah:array[1..10] of character, top:integer>

Function isFullStack(S:Tstack) -> boolean
{mengembalikan true bila S penuh}

kamus lokal
full : boolean

cek lagi
ya gais ga
tau bener
ato ga



Algoritma

```
if S.top = 10 then
    full <- true
else
    full <- false
-> full
```

Type Tstack = <wadah:array[1..10] of character, top:integer>

Procedure Push(input/output S:Tstack, input e:character)

{I.S: S,e terdefinisi, S mungkin kosong }

{F.S: S tetap, atau infotop(S)=e }

{Proses mengisi elemen e ke puncak S, bila belum penuh}

KAMUS LOKAL

ALGORITMA

if S.top < 10 then

S.top <- S.top + 1

S.wadah[S.top] <- e;

if S.top < 10 then

if NOT isFullstack(S) then

Type Tstack = <wadah:array[1..10] of character, top:integer>

Procedure Pop(input/output S:Tstack, output e:character)

{I.S: S terdefinisi, mungkin kosong }

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

{Proses menghapus elemen e dari puncak S, bila belum kosong}

KAMUS LOKAL

ALGORITMA

```
if S.top <= 0 then
    e <- S.wadah[S.top] { mengeluarkan nilai teratas(Top) ke variable e }
    S.wadah[S.top] <- ' ' { mengosongkan wadah top }
    S.top <- S.top - 1 { membuat posisi top ke posisi sebelum nya}
else { S.top = 0 }
    e <- ' ' { mengembalikan char kosong bila S.top = 0 }
```

kalau elemen puncak ga kosong,
dikosongin. Kalau udah
biarin aja itu. Sagi outputnya (tetep
kosong)

```
if S.top <= 0 then
    e <- S.wadah[S.top]
    S.wadah[S.top] <- ' '
    S.top <- S.top - 1
else { S.top = 0 }
    e <- ' '
    S.top <- S.top
```

Type Tstack = <wadah:array[1..10] of character, top:integer>

Procedure PrintStack(input S:Tstack)
{I.S:-; F.S:-; Proses: menampilkan info elemen S }

kamus lokal
i : integer

algoritma
i traversal 1..10
 output S.wadah[i]

Procedure addX (input/output T:Tabel, input X: integer)
 {I.S.: T terdefinisi, X terdefinisi }
 {F.S.: isi T.wadah bertambah 1 elemen jika belum penuh}
 {Proses: mengisi elemen T.wadah dengan nilai X}

Kamus Lokal :
 i : integer {iterator}

Algoritma:
if T.size != 10 then
 T.size <- T.size+1
 T.wadah[T.size] <- X

