```
function IsJValid (HH,MM,SS:integer) -> boolean
{ Mengirim true jika HH,MM,SS dpt membentuk J valid, untuk mentest sebelumnya }
Kamus lokal

Algoritma
   if ( (HH<0 or HH>24 ) or (MM <0 or MM > 59) or (SS<0 or SS>59) then
        -> False
   else
        -> True

function MakeJam ( HH, MM, SS : integer ) -> JAM
{ Membentuk sebuah JAM dari komponen yg valid }
Kamus lokal
     function IsJValid (HH,MM,SS: integer) -> boolean
     J : JAM
Algoritma
     if IsJValid (HH,MM,SS) then
          J.HH <- HH
          J.MM <- MM
          J.SS <- SS
     else
          J.HH <- 99
          J.MM <- 99
          J.SS <- 99
     -> J

function GetHour (J : JAM) -> integer
{ Mengirim bagian HH dari JAM }
Kamus lokal

Algoritma
 -> J.HH

function GetMinute (J : JAM) -> integer
{ Mengirimkan bagian MM dari JAM }
Kamus lokal

Algoritma
   -> J.MM
```

Scanned by CamScanner

```
function GetSecond (J : JAM) → integer
{ Mengirimkan bagian SS dari JAM }
Kamus lokal

Algoritma
    → J.SS

procedure SetH (input/output J : JAM, input newH : integer )
{ Mengubah nilai komponen HH dari J }
Kamus lokal

Algoritma
    input (newHH)
    J.HH ← newHH

procedure SetM (input/output J : JAM, input newM : integer)
{ Mengubah nilai komponen MM dari J }
Kamus lokal

Algoritma
    input (new MM)
    J.MM ← newMM

procedure SetS (input/output J : JAM, input newS : integer)
{ Mengubah nilai komponen SS dari J }
Kamus lokal

Algoritma
    input (new SS)
    J.SS ← new SS
```

Scanned by CamScanner

```
procedure BacaJam (output J: Jam)
{IS : Jam tidak terdefinisi}
{F.S : J terdefinisi dan merupakan jam yg valid}
{Proses : mengulangi membaca komponen H, M, S sehingga membentuk J yg
 valid. Tidak mungkin menghasilkan J yang tidak valid}
Kamus lokal
    function isJValid (HH, MM, SS : integer) -> boolean
Algoritma
    if isJValid(HH, MM, SS : integer) then
        -> True
    else
        -> False

procedure TulisJam (input J : JAM)
{ I.S. : J sembarang }
{ F.S. : Nilai J ditulis dengan format HH : MM:SS }
{ Proses: menulis nilai setiap komponen J ke layar }
Kamus lokal
    procedure BacaJam (output J : JAM)

Algoritma
    output (procedure BacaJam (output J : JAM))

function JamToDetik (J : JAM) -> integer
{ Mengubah Jam ke detik }
Kamus lokal

Algoritma
    -> (3600 * J.HH) + (60 * J.MM) + (J.SS)

function DetikToJam ( N : integer) -> JAM
{ Mengubah detik ke bentuk JAM }
Kamus lokal
    J : JAM
Algoritma
        J.HH <- N mod 3600
        J.MM <- (N div 3600) mod 60
        J.SS <- (N div 3600) div 60
```