```
Program NilMax2Tabel
{deskripsi blabla}

Kamus
      T: array [1..100] of integer
      M1: integer
      M2: integer
      i: integer {counter traversal}
Algoritma
      M1 ← 0; M2 ← 0
      i traversal [1..100]
            input(Ti)
            depend on Ti, M1, M2:
                  Ti >= M1: M1 ← Ti
                  Ti > M2: M2 ← Ti
      {endTraversal}
      output(M2)
```

```
TabInt:
→ Neff: integer
→ TI: array [1..100] of integer

Function NBElmt (T: TabInt) → integer

Kamus Lokal
     -

Algoritma
     → T.Neff

Function IdxLast (T: TabInt) → integer

Kamus Lokal
     -

Algoritma
     → T.TI_NBElmt(T)

Function IsEq (T1: TabInt, T2: TabInt) → boolean

Kamus Lokal
     i: integer
     beda: boolean

Algoritma
     if (T1.Neff = T2.Neff) then
          beda ← true
          i traversal [1..(T1.Neff)]
               if (T1_i /= T2_i) then
                    beda ← false
          {endTraversal}
     else
          → false

     → beda
```

```
Function SearchIdx (T:TabInt, X:ElmtType) → IdxType

Kamus Lokal
      i: IdxType
      ketemu: IdxType
Algoritma
      ketemu ← -999
      i ← 1
      while (ketemu = -999 AND i <= T.Neff) do
            if (Ti = X) then
                  ketemu ← i
            i ← i + 1
      {endWhile}
      → ketemu

Function SearchBool (T:TabInt, X:ElmtType) → boolean

Kamus Lokal
      i: IdxType
      ketemu: boolean
Algoritma
      ketemu ← false
      i ← 1
      while (ketemu = false AND i <= T.Neff) do
            if (Ti = X) then
                  ketemu ← true
            i ← i + 1
      {endWhile}
      → ketemu

Procedure CopyTab(input: Tin:TabInt, output: Tout:TabInt)

Kamus Lokal
      N: integer
      Tout: TabInt
      i: integer

Algoritma
      N ← Tin.Neff {asumsikan Tin terisi penuh}
      Tout.Neff ← N
      i traversal [1..N]
            Touti ← Tini
      {endTrav}
```

```
Function IsSimetris  (T: TabInt) → boolean

< 2, 4, 5, 4, 2 > → FALSE
< 2, 4, 3, 3, 4, 1 > → FALSE
< 2, 4, 3, 3, 4, 2 > → TRUE

Kamus Lokal
     i: integer
     j: integer
     simetris: boolean
Algoritma
     simetris ← true
     if (T.Neff mod 2 /= 0) then
          simetris ← false
     else
          i traversal [1..((T.Neff)/2)]
               if (T.TIi /= T.TIN-i+1) then
                    simetris ← false
          {endTraversal}

     → simetris

Program IsiMatrik
{isi deskripsi}

Kamus
     M: integer
     N: integer
     i: integer
     j: integer
     Matriks: array [1..M][1..N] of integer

Algoritma
     input(M, N)
     if (M /= N || M < 0 || N < 0) then
          output("Nilai M dan N harus sama")
     else
          i traversal [1..M]
               j traversal [1..N]
                    depend on i,j:
                         j < i: Matriks_i,j ← 2
                         j > i: Matriks_i,j ← 0
                         j = i: Matriks_i,j ← 1
```