**pseudocode**

**MaxHeapify(A, n, i )**

**//Array A of size n and root node of index i is passed**

largest ← i

left ← 2 \* i + 1

right ← 2 \* i + 2

if left < n and A[largest] < A[left]

then largest ← left

if r < n and A[right] > A[largest]

then largest ← right

if largest ≠ i

then swap A[i] ↔ A[largest]

MaxHeapify (A, n, largest)

**Build\_Max\_Heap(A)**

**//Array A is passed**

**n** ← size[A]

for i ← ⎣*n*/2⎦ downto 1 do

MaxHeapify (A, n, i)

**Algorithm HeapSort (A)**

**//Array A is passed**

**n** ← size[A]

Build-Max-Heap(A, n)

for (i ← n-1) downto 0 do

swap A[0] ↔ A[i ]

MaxHeapify (A,n, 0)

**Alogrithm isTriangular(A,n)**

HeapSort (A)

for i ← to n-2 do

if A[i] + A[i + 1] > A[i + 2]

then return 1

else

return 0

**Analysis**

