Heap Implementation with Full Trees

Idea: A full binary tree in the array $H[0]...H[n-1]$, where an item is always $\leq$ its children (the root has the smallest element). $nextRB$, which is initially 0, denotes the position directly after that of the RB leaf (i.e., $nextRB = RB+1$):

INSERT: Place the new item after the RB leaf and "percolate up":

procedure PERCUP($i$):
    while $i > 0$ and $H[i] < H[PARENT(i)]$ do begin
        exchange $H[i]$ with $H[PARENT(i)]$
        $i := PARENT(i)$
    end
end

procedure INSERT($d$):
    if $nextRB \geq n$ then ERROR — heap is full
else begin
    $H[nextRB] := d$
    PERCUP(nextRB)
    nextRB := nextRB + 1
end
end
(heap implementation with full trees, continued)

**DELETEMIN:** Exchange the root with the RB leaf and "percolate down":

```plaintext
procedure PERCDOWN(i)
  while i is not a leaf and \( H[i] \) is greater than one of its children do begin
    Determine the child \( j \) of \( i \) such that \( H[j] \) is the smallest.
    Exchange \( H[i] \) with \( H[j] \).
    \( i := j \)
  end
end
```

**function DELETEMIN:**

```plaintext
if nextRB \leq 0 then ERROR — heap is empty
else begin
  nextRB := nextRB - 1
  exchange \( H[0] \) and \( H[nextRB] \)
  PERCDOWN(0)
return \( H[nextRB] \)
end
```

*Note:* Deleted element is left in \( H[nextRB] \), the position after the new RB leaf.