Algorithms (COT 6405): Solutions 7

Problem 1
Let $A[1..n]$ be a sorted array of n distinct integer numbers. Write an efficient algorithm INDEX-SEARCH(A, n) that finds an index i such that $A[i] = i$.

The algorithm is almost identical to BINARY-SEARCH, and its time complexity is $O(\lg n)$. It works only for integer arrays, since it is based on the assumption that, for every two indices p and r (where $p \leq r$), we have $A[r] - A[p] \geq r - p$.

INDEX-SEARCH(A, n)
$p \leftarrow 1$
$r \leftarrow n$
while $p < r$
  do $q = \lfloor (p + r)/2 \rfloor$
    if $q \leq A[q]$
      then $r \leftarrow q$
      else $p \leftarrow q + 1$
if $p = A[p]$
  then return p
  else return 0

Problem 2
We consider an array $A[1..n]$ and define a segment sum from p to r:

$$sum(p, r) = \sum_{p \leq i \leq r} A[i].$$

Write a linear-time algorithm that determines the maximum over all segment sums.

MAX-SEGMENT(A, n)
Local-Max $\leftarrow A[1]$
Global-Max $\leftarrow A[1]$
for $i \leftarrow 2$ **to** n
  do *Local-Max* $\leftarrow \max(A[i], \textit{Local-Max} + A[i])$
        $\triangleright$ *Local-Max* is the maximum over the segments whose last element is $A[i]$
    Global-Max $\leftarrow \max(\textit{Local-Max}, \textit{Global-Max})$
        $\triangleright$ *Global-Max* is the maximum over all segments in $A[1..i]$
return *Global-Max*