9.3-8

Let's start out by supposing that the median (the lower median, since we know we have an even number of elements) is in X. Let's call the median value m, and let's suppose that it's in $X[k]$. Then k elements of X are less than or equal to m and $n - k$ elements of X are greater than or equal to m. We know that in the two arrays combined, there must be n elements less than or equal to m and n elements greater than or equal to m, and so there must be $n - k$ elements of Y that are less than or equal to m and $n - (n - k) = k$ elements of Y that are greater than or equal to m. Thus, we can check that $X[k]$ is the lower median by checking whether $Y[n-k] \le X[k] \le Y[n - k + 1]$. A boundary case occurs for $k = n$. Then $n - k = 0$, and there is no array entry $Y[0]$; we only need to check that $X[n] \le Y[1]$.

Now, if the median is in X but is not in $X[k]$, then the above condition will not hold. If the median is in $X[k']$, where $k' < k$, then $X[k]$ is above the median, and $Y[n - k + 1] < X[k]$. Conversely, if the median is in $X[k'']$, where $k'' > k$, then $X[k]$ is below the median, and $X[k] < Y[n - k]$.

Thus, we can use a binary search to determine whether there is an $X[k]$ such that either $k < n$ and $Y[n-k] \le X[k] \le Y[n-k+1]$ or $k = n$ and $X[k] \le Y[n-k+1]$; if we Þnd such an $X[k]$, then it is the median. Otherwise, we know that the median is in Y, and we use a binary search to find a $Y[k]$ such that either $k < n$ and $X[n - k] \le Y[k] \le X[n - k + 1]$ or $k = n$ and $Y[k] \le X[n - k + 1]$; such a $Y[k]$ is the median. Since each binary search takes $O(\lg n)$ time, we spend a total of $O(\lg n)$ time.

Here's how we write the algorithm in pseudocode:

TWO-ARRAY-MEDIAN(X, Y)

$n \leftarrow$ *length*[X] ◇ n also equals *length*[Y]

median $\leftarrow$ FIND-MEDIAN(X, Y, n, 1, n)

if *median* **=** NOT-FOUND

then *median* $\leftarrow$ FIND-MEDIAN(Y, X, n, 1, n)

return *median*

FIND-MEDIAN(A, B, n, *low*, *high*)

if *low* > *high*
then return NOT-FOUND
else $k \leftarrow \lfloor (low+high)/2 \rfloor$ **if** $k = n$ and $A[n] \le B[1]$
then return $A[n]$
elseif $k < n$ and $B[n-k] \le A[k] \le B[n-k+1]$
then return $A[k]$
elseif $A[k] > B[n-k+1]$
then return FIND-MEDIAN($A, B, n, low, k-1$)
else return FIND-MEDIAN($A, B, n, k+1, high$)

15.4-5

作法:

假設 input numbers 爲 X
Step 1: 將 X 複製一份到 Y
Step 2: 對 Y 做 Merge Sort
Step 3: 對 X, Y 做 LCS
Step 4: 利用 PRINT-LCS 把結果 output 出來

說明:

因爲這題要我們找 longest monotonically increasing subsequence, 所以我們可以利用 Y, Y 由 X 複製來的, 所以

(1) Y 的元素和 X 相同.
(2) Y 是 sort 好的,所以滿足 monotonically increasing
(3) X, Y 的 LCS 是 X 和 Y 的 longest common subsequence, 所以是 X 的 subsequences 中最長且滿足 monotonically increasing 性質的

時間分析:

Step 1: 複製 n 個數→ $O(n)$
Step 2: Merge sort → $O(n \lg n)$
Step 3: 做 LCS, X, Y 的長度都是 n, → $O(n^2)$
Step 4: 做 PRINT-LCS → $O(n)$
所以總共的時間爲 $O(n^2)$