P4

Prompt:

Given an array a_1, ..., a_n of n distinct integers, design an O(log n) time algorithm that finds a_i such that $a_i > a_{i-1}$ and $a_i > a_{i+1}$.

Algorithm:

```
// Given array A and range indices lo and hi, find a local max a_i
LocalMax(A, lo, hi):
    // Handle the base cases
    If hi − lo < 0 then
        Return "Impossible"
    Else if hi − lo = 0 then
        Return A[lo]
    EndIf

    // Check the upper and lower bounds of A[lo:hi]
    If A[lo] > A[lo+1] then
        Return A[lo]
    Else if A[hi] > A[hi−1] then
        Return A[hi]
    EndIf

    // Check the middle index of A and recurse
    // Let mid always be an integer
    Let mid = lo + (hi − lo) / 2
    If A[mid] > A[mid−1] and A[mid] > A[mid+1] then
        Return A[mid]
    Else if A[mid−1] > A[mid+1] then
        Let hi = mid − 1
        Return LocalMax(A, lo, hi)
    Else then
        Let lo = mid + 1
        Return LocalMax(A, lo, hi)
    EndIf
```

Claim:

The algorithm terminates in O(log n) time.

Proof:

The algorithm reduces the problem into a variant of binary search which is known to terminate in O(log n). That is, it solves a problem of size n by reducing it into one subproblem of half the size, which it recursively solves, and combines each level in constant time. We can express the runtime of the algorithm as a recurrence relation of the form $T(n) \leq T(\lceil \frac{n}{2} \rceil) + O(1)$ when $n > 1$, and $T(1) \leq c$. Which, by the master theorem is O(log n). $\square$

Claim:

Given an array a_1, ..., a_n of n distinct integers, the algorithm finds an a_i such that $a_i > a_{i-1}$ and $a_i > a_{i+1}$. If $a_1 > a_2$ or $a_n > a_{n-1}$ the algorithm returns a_1 or a_n, respectively.

Proof:

Let P(n) be the claim that, given an array $A = a_1$, ..., a_n of n distinct integers, if $a_1 > a_2$ or $a_n > a_{n-1}$ the algorithm returns a_1 or a_n, respectively, otherwise, the algorithm returns an a_i such that $a_i > a_{i-1}$ and $a_i > a_{i+1}$. Prove P(n) by induction.

Base Case: P(1) holds because $a_1 = a_i = a_n$ is the maximum value by dint of being the only value.

Inductive Hypothesis: For some $k \geq 1$, assume P(j) holds for $1 \leq j \leq k-1$.

Inductive Step: Goal, show P(k) holds. Let indices hi and lo represent the upper and lower bounds of A such that $k = hi - lo$. If, by simple comparison, $a_{lo} > a_{lo+1}$ or $a_{hi} > a_{hi-1}$ a local maximum, a_{lo} or a_{hi}, is immediately found and we are done. If not, let index $mid = lo - \frac{hi-lo}{2}$. If $a_{mid} > a_{mid-1}$ and $a_{mid} > a_{mid+1}$ then a_{mid} is a local maximum and we are done. Otherwise, we check either the range to the left or right of mid. If $mid - 1 > mid + 1$, check the range between lo and $mid - 1$ for a maximum. Because $(mid - 1) - lo = (lo + \frac{hi-lo}{2} - 1) - lo = \frac{k}{2} - 1 < k$, P$(\frac{k}{2} - 1)$ holds. Or if $mid - 1 \leq mid + 1$, check the range between $mid + 1$ and hi for a maximum. Because $hi - (mid + 1) = hi - (lo + \frac{hi-lo}{2}) - 1 = \frac{k}{2} - 1 < k$, P$(\frac{k}{2} - 1)$ holds. $\square$