CMPS 102 — Fall 2018 – Homework 2

Alyssa Melton
I have read and agree to the collaboration policy.
Collaborators: none

Solution to Problem 2

Important definitions from the problem:
$A[1] \leq A[2]$
$A[n-1] \geq A[n]$
An element $A[x]$ is a peak if $A[x] \leq A[x-1]$ and $A[x] \geq A[x+1]$

It's important to note that $A[1] \leq A[2]$ and $A[n-1] \leq A[n]$. This implies that there $must$ be a peak somewhere between the first and last element. Even if $A[1] = A[n]$, by definition of a peak, all of the elements between 1 and n can be a peak if they're all the same number.

Using these facts; I've come up with the following algorithm:

$i =$ beginning of Array(or subarray)
$j =$ end of Array (or subarray)
findpeak (A, i, j)
  let middle $= i + \frac{j-i}{2}$

  if $A[middle-1] \leq A[middle]$ and $A[middle] \geq A[middle+1]$
  $\rightarrow$ return $A[middle]$.
  if $A[middle-1] > A[middle]$ (there must be a peak to the left)
  $\rightarrow$ return findpeak $(A, i, middle-1)$.
  if $A[middle] < A[middle+1]$ (there must be a peak to the right)
  $\rightarrow$ return findpeak $(A, middle+1, j)$.

Claim 1. *This algorithm will always terminate and return a peak.*

Proof. Assume the algorithm did not return a peak. Then, there must not be an element $k \in \{2, 3, ..., n-1\}$ that is a peak, such that $A[k-1] \leq A[k]$ and $A[k] \geq A[k+1]$.

Then, one of two cases must hold:
Case 1: The array is constantly increasing, such that $\forall A[m]$, $m \in \{2, 3, ..., n\}$, $A[m-1] < A[m]$. But consider the case when $A[m]$ is the second to last element in the array, namely, $A[n-1]$. By definition, $A[n-1] \geq A[n]$, which is a contradiction. Then, $A[n-1]$ is a peak and it would be returned. Thus, it must be the case that $\exists A[m]$, $m \in \{1, 2, ..., n\}$ such that $A[m-1] \geq A[m]$.

Case 2: The array is constantly decreasing, such that $\forall A[m]$, $m \in \{2, 3, ..., n\}$, $A[m-1] > A[m]$. But consider the case when $A[m]$ is the second element in the array, namely, $A[2]$. By definition, $A[1] \leq A[2]$, which is a contradiction. Then, $A[2]$ is a peak, and it would be returned. Thus, it must be the case that $\exists A[m]$, $m \in \{1, 2, ..., n\}$ such that $A[m-1] \geq A[m]$.

There are no other cases to consider, because all other cases would include a peak, so we can conclude that in all cases, a peak exists and will be returned.

□

Claim 2. *This algorithm runs in $O(log\ n)$ time.*

Proof. Each recurrance, we are performing findpeak (A, i, j) on a subarray that is at most half the length of the original array, and are doing two comparisons. We get the following recurrence relation:
$T(n) = T(\frac{n}{2}) + 2$ until $n = 1$.
$\implies T(\frac{n}{2^k}) + 2k$
$\implies 1 = \frac{n}{2^k}$
$\implies 2^k = n \implies k = log_2 n. \implies O(log\ n)$ □