In an attempt to improve the worst-case computational complexity of QUICKSORT, the PARTITION algorithm we have seen in class was modified as follows.
For the new algorithm, assume that all the elements in the array are different, and there are three or more elements in the array.
The new algorithm finds the smallest element, \( A[m_1] \), such that \( p \leq m_1 \leq r \).
It also finds the largest element, \( A[m_2] \), such that \( p \leq m_2 \leq r \).
It then finds an index \( i \) such that \( p \leq i \leq r \), and \( A[m_1] < A[i] < A[m_2] \).
The new algorithm exchanges \( A[i] \) with \( A[r] \), and applies the PARTITION algorithm we have seen in class.

Analyze QUICKSORT with the new PARTITION algorithm and determine whether or not it improves the worst-case computational complexity.

Answer:

In the worst case PARTITION will produce two subarrays of sizes 1 and \( n-2 \).
The corresponding recurrence for QUICKSORT is:
\[
T(n) = T(n-2) + T(1) + \Theta(n) = T(n-2) + \Theta(n).
\]
Solving for \( T(n) = T(n-2) + n \):
\[
T(n) = T(n-2) + n = T(n-4) + (n-2) + n = T(n-6) + (n-4) + (n-2) + n = \ldots = \sum_{i=0}^{n/2} (n-2i) = \sum_{i=0}^{n/2} n - 2 \sum_{i=0}^{n/2} i = n^2/2 - n^2/4 = n^2/4 = O(n^2).
\]

There is no improvement in the worst-case complexity.