Algorithms (COT 6405): Solutions 3

Problem 1

Write an algorithm that combines INSERTION-SORT and MERGE-SORT.

The following algorithm calls INSERTION-SORT for array segments whose length is at most k; the running time of this algorithm is $\Theta(n \cdot k + n \cdot \lg(n/k))$.

```
INSERTION-SORT(A, p, r)
for j ← p + 1 to r
    do key ← A[j]
       i ← j − 1
       while i ≥ p and A[i] > key
           do A[i + 1] ← A[i]
              i ← i − 1
       A[i + 1] ← key
```

```
COMBINED-SORT(A, p, r, k)
if r − p < k
    then INSERTION-SORT(A, p, r)
    else q ← ⌊(p + r)/2⌋
         COMBINED-SORT(A, p, q, k)
         COMBINED-SORT(A, q + 1, r, k)
         MERGE(A, p, q, r)
```

Problem 2

Write an algorithm that inputs an integer array $A[1..n]$ and an odd integer number k, and determines whether k can be represented as the sum of two elements of the array.

The following algorithm uses two subroutines, MERGE-SORT (described in class) and BINARY-SEARCH (described in Assignment 2); the running time of this algorithm is $O(n \cdot \lg n)$.

```
SUM-SEARCH(A, n, k)
MERGE-SORT(A, 1, n)   ▷ sort the array A[1..n]
for i ← 2 to n
    do if BINARY-SEARCH(A, n, k − A[i]) ≠ 0
          then return TRUE   ▷ k is the sum of two elements of A[1..n]
return FALSE   ▷ k is not the sum of any two elements
```