Algorithms (COT 6405): Solutions 2

Problem 1

Write an algorithm that finds a given value k in a sorted array $A[1..n]$. It should return the index of the found element; if the array does not include k, it should should return 0.

```
BINARY-SEARCH($A, n, k$)
$p \leftarrow 1$
$r \leftarrow n$
while $p < r$
   do $q = \lfloor (p+r)/2 \rfloor$
       if $k \leq A[q]$
          then $r \leftarrow q$
          else $p \leftarrow q+1$
if $k = A[p]$
   then return $p$
   else return 0
```

Problem 2

Let $A[1..n]$ be an array of n distinct numbers. If $i < j$ and $A[i] > A[j]$, then the pair (i, j) is called an *inversion*. Write an algorithm that determines the number of inversions in $A[1..n]$.

```
INVERSIONS($A, n$)
$counter \leftarrow 0$
for $i \leftarrow 1$ to $n-1$
    do for $j \leftarrow i+1$ to $n$
           do if $A[i] > A[j]$
                  then $counter \leftarrow counter + 1$
return $counter$
```

The time complexity of the INVERSIONS algorithm is $\Theta(n^2)$.