Section 7.3

7.3.1

```
CountValuesLessThanT

Input: a1, a2,...,an
  n, the length of the sequence.
  T, a target value.
Output: The number of values in the sequence that are less than T.

count := 0

For i = 1 to n
  If ( ai < T ), count := count + 1
End-for

Return( count )
```

a. *Characterize the asymptotic growth of the worst-case time complexity of the algorithm. Justify your answer.*

Proof. For any input of size n, the loop in the algorithm will execute n times, which is at worst n. Therefore the number of operations in the worst case is $cn + d$, which is $\mathcal{O}(n)$. □

7.3.2

```
MaximumSubsequenceSum

Input: a1, a2,...,an
  n, the length of the sequence.
Output: The value of the maximum subsequence sum.

maxSum := 0

For i = 1 to n
  thisSum := 0
  For j = i to n
    thisSum := thisSum + aj
    If ( thisSum > maxSum ), maxSum := thisSum
  End-for
End-for

Return( maxSum )
```

a. *Characterize the asymptotic growth of the worst-case time complexity of the algorithm. Justify your answer.*

Proof. For any input of size n, the outer loop in the algorithm will execute n times, which is at worst n. The inner loop will execute $n - j$ times, which is at worst n. Therefore the number of operations in the worst case is $cn^2 + dn + f$, which is $\mathcal{O}(n^2)$. □

b. *Can you find an algorithm that solves the same problem whose worst-case time complexity is linear?*

```
MaximumSubsequenceSumLinear

Input: a1, a2,...,an
  n, the length of the sequence.
  Output: The value of the maximum subsequence sum.

maxSum := a1
thisSum := a1

For i = 2 to n
  thisSum := max( ai, thisSum + ai )
  maxSum := max( maxSum, thisSum )
End-for

Return( maxSum )
```

7.3.3

```
FindMaxFunctionValue

Input: a1, a2,...,an
  n, the length of the sequence.
Output: The largest values of M on input values from the sequence.

max := M(a1, a1, a1)

For i = 1 to n
  For j = 1 to n
    For k = 1 to n
      new := M(ai, aj, ak)
      If ( new > max ), max := new
    End-for
  End-for
End-for

Return( max )
```

a. *Characterize the asymptotic growth of the worst-case time complexity of the algorithm. Justify your answer.*

Proof. For any input of size n, the outer loop in the algorithm will execute n. The middle loop will execute n times. The innermost loop will execute n times. Therefore the number of operations in the worst case is $an^3 + bn^2 + cn + d$, which is $\mathcal{O}(n^3)$. □