CSE 331 Algorithms and Data Structures
Homework 1 Solution

solution

1. Rewrite the INSERTION-SORT procedure to sort into nonincreasing instead of nondecreasing order. Using the same pseudocode conventions as in the lecture.

   Let $A[1 \ldots n]$ be an array of numbers. Array A's length is length[A]
   INSERTION-SORT(A).

   ```
   1 for j ← 2 to length[A]
   2       do temp ← A[j]
   3            Insert A[j] into the sorted sequence A[1 : j − 1]
   4          i ← j − 1
   5          whi le i > 0 and A[i] < temp
   6                do A[i+1] ← A[i]
   7                   i ← i − 1
   8          A[i+1] ← temp
   ```

2. Consider the following functions in terms of order of magnitude. Order (list) them in terms of the growth, such that slow growth functions are listed before fast growth functions. That is, if f is listed before g, $f = O(g)$. Also, partition the functions so that if $f = \Theta(g)$, then they are in the same class.

   2/N
   37
   $\log(N)$
   $\sqrt{N}$

N
$N \log N$, $N \log(N^2)$
$N \log^2(N)$
$N^{1.5}$
N^2
$N^2 \log(N)$
N^3
2^N
$N!$

Note that $\log(N^2)$ is $2 \log(N)$.

3. Prove that $3n^3 + 2n \log n + 10 = O(n^3)$.

   Proof: we need to find a constant c and n0 such that $3n^3 + 2n \log n + 10 \leq cn^3$ for all $n \geq n0$.
   Let $c = 15$ and $n0 = 1$. According to the relationship of the commonly used functions, we have:

   $$\begin{aligned} 15n^3 &= 3n^3 + 2n^3 + 10n^3 \\ &\geq 3n^2 + 2n \log n + 10 \text{ for } n \geq 1 \end{aligned}$$

4. Suppose $T_1(N) = O(f(N))$ and $T_2(N) = O(f(N))$. Evaluate each statement below. If it is true, prove it. If it is not, provide a counter-example.

   - a. $T_1(N) + T_2(N) = O(f(N))$
     We have $T_1(N) \leq c_1 f(N)$ when $N \geq n_1$ and $T_2(N) \leq c_2 f(N)$ when $N \geq n_2$. Apparently $T_1(N) + T_2(N) \leq (c_1 + c_2) f(N)$ when $N \geq \max(n_1, n_2)$. So $T_1(N) + T_2(N) = O(f(N))$.
   - b. $T_1(N)/T_2(N) = O(1)$
     Assume $T_1(N) = n^2$ and $T_2(N) = n$. According to Definition 2.1, $T_1(N) = O(n^2)$ and $T_2(N) = O(n^2)$. But $T_1(N)/T_2(N) = n$, which is $O(n)$. So the statement is false.
   - c. $T_1(N) = O(T_2(N))$
     Still use the above example, $T_1(N) = O(n^2)$, not $O(n)$. So the statement is false.

5. Let $f(n)$ and $g(n)$ be asymptotically nonnegative functions. Using the basic definition of Θ-notation, prove that $max(f(n), g(n)) = \Theta(f(n) + g(n))$.

   We have: $1/2((f(n) + g(n)) \leq max(f(n), g(n)) \leq (f(n) + g(n))$.
   Here, $c1 = 1/2$ and $c2 = 1$ and $n0 = 0$. Note that the above proof works for $max(f(n), g(n)) = \Theta(f(n) + g(n))$

6. For each of the following four program fragments, give an analysis of the running time (Big-O will do).

   (a)
   ```
   Sum = 0;
   for i = 1 to N do
       Sum = Sum +i;
   ```
   Answer: O(N)

   (b)
   ```
   Sum = 0;
   for i = 1 to N do
       for j = 1 to N do
           for k = 1 to N do
               Sum = Sum + 1;
   ```
   Answer: $O(N^3)$

   (c)
   ```
   Sum = 0;
   for i = 1 to N^2 do
       for j = 1 to N do
           Sum = Sum + 1;
   ```
   Answer: $O(N^3)$