Assignment 1 (Sept. 8th 2016)

Due date and time: Sept 21st 17:00p.m.
Submit in class (handwritten hardcopy).

For the first Assignment, you must work alone.

Following are problems taken from the Book:

1-3. [5] Design/draw a road network with two points $a$ and $b$ such that the fastest route between $a$ and $b$ is not the shortest route.

1-5. [4] The knapsack problem is as follows: given a set of integers $S = \{s_1, s_2, \ldots, s_n\}$, and a target number $T$, find a subset of $S$ which adds up exactly to $T$. For example, there exists a subset within $S = \{1, 2, 5, 9, 10\}$ that adds up to $T = 22$ but not $T = 23$.

Put the elements of $S$ in the knapsack in left to right order if they fit, i.e. the first-fit algorithm.

1-9. [3] Prove the correctness of the following sorting algorithm.

$$
function \text{ bubblesort} (A: \text{list}[1 \ldots n])
\begin{align*}
&\text{var int } i, j \\
&\text{for } i \text{ from } n \text{ to } 1 \\
&\quad \text{for } j \text{ from } 1 \text{ to } i - 1 \\
&\quad\quad \text{if } (A[j] > A[j + 1]) \\
&\quad\quad\quad \text{swap the values of } A[j] \text{ and } A[j + 1]
\end{align*}
$$

1-14. [5] Prove by induction on $n \geq 1$ that for every $a \neq 1$,

$$
\sum_{i=0}^{n} a^i = \frac{a^{n+1} - 1}{a - 1}
$$

1-28. [5] Write a function to perform integer division without using either the / or * operators. Find a fast way to do it.

2-1. [3] What value is returned by the following function? Express your answer as a function of $n$. Give the worst-case running time using the Big Oh notation.

$$
function \text{ mystery}(n)
\begin{align*}
&r := 0 \\
&\text{for } i := 1 \text{ to } n - 1 \text{ do} \\
&\quad \text{for } j := i + 1 \text{ to } n \text{ do} \\
&\quad\quad \text{for } k := 1 \text{ to } j \text{ do} \\
&\quad\quad\quad r := r + 1 \\
&\text{return}(r)
\end{align*}
$$
2-5. [5] Suppose the following algorithm is used to evaluate the polynomial

\[ p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 \]

(a) How many multiplications are done in the worst-case? How many additions?
(b) How many multiplications are done on the average?
(c) Can you improve this algorithm?

2-9. [3] For each of the following pairs of functions \( f(n) \) and \( g(n) \), determine whether \( f(n) = O(g(n)) \), \( g(n) = O(f(n)) \), or both.

(a) \( f(n) = (n^2 - n)/2 \), \( g(n) = 6n \)
(c) \( f(n) = n \log n \), \( g(n) = n^{1/2}/2 \)
(f) \( f(n) = 4n \log n + n \), \( g(n) = (n^2 - n)/2 \)

2-10. [3] Prove that \( n^3 - 3n^2 - n + 1 = \Theta(n^3) \).

2-20. [5] Find two functions \( f(n) \) and \( g(n) \) that satisfy the following relationship. If no such \( f \) and \( g \) exist, write “None.”

(a) \( f(n) = o(g(n)) \) and \( f(n) = \Theta(g(n)) \)
(c) \( f(n) = \Theta(g(n)) \) and \( f(n) = O(g(n)) \)

2-23. [3] For each of these questions, briefly explain your answer.
(a) If I prove that an algorithm takes \( O(n^3) \) worst-case time, is it possible that it takes \( O(n) \) on some inputs?
(b) If I prove that an algorithm takes \( O(n^2) \) worst-case time, is it possible that it takes \( O(n) \) on all inputs?
(c) If I prove that an algorithm takes \( \Theta(n^2) \) worst-case time, is it possible that it takes \( O(n) \) on some inputs?