CSE 373: Analysis of Algorithms

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, . . . , *sn*}, 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.
Find counterexamples to each of the following algorithms for the knapsack problem. That is, giving an *S* and *T* such that the subset is selected using the algorithm does not leave the knapsack completely full, even though such a solution exists.

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 bubblesort (A : list[1 ... n])
    var int i, j
    for i from n to 1
        for j from 1 to i − 1
            if (A[j] > A[j + 1])
                swap the values of A[j] and A[j + 1]
```

1-14. *[5]* Prove by induction on $n \geq 1$ that for every *a* != 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 mystery(n)
    r := 0
    for i := 1 to n − 1 do
        for j := i + 1 to n do
            for k := 1 to j do
                r := r + 1
    return(r)
```

2-5. *[5]* Suppose the following algorithm is used to evaluate the polynomial

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

```
p := a_0;
xpower := 1;
for i := 1 to n do
        xpower := x * xpower;
        p := p + a_i * xpower
end
```

(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) $f(n) = n \log n$, $g(n) = n\sqrt{n}/2$

(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 $O(n^2)$ 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?