Ally Smith

Homework 2

1. True or false?

   (a) Is $2^{n+1} = O(2^n)$? **True**

   (b) Is $2^{2n} = O(2^n)$? **False**

2. Answer the following

   (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?

   Yes. For example, a sorting algorithm given a list in reverse order may take $O(n^2)$, while when provided a sorted list it is only $O(n)$.

   (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?

   Yes. Because Big Oh notation offers an upper limit, it is possible that the proven $O(n^2)$ is true while also being true that it is also $O(n)$.

   (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?

   Yes. Because it is $\Theta(n^2)$ of the worst case, it is reasonable to assume that there could be inputs that are $O(n)$.

   (d) If I prove that an algorithm takes $\Theta(n^2)$ worst-case time, is it possible that it takes $O(n)$ on all inputs?

   No. Because the algorithm is $\Theta(n^2)$ there must be an input that takes $O(n^2)$.

3. Assume n is even. Let $T(n)$ denote the number of times 'foobar' is printed as a function of n.

   (a) Express $T(n)$ as three nested summations.

   $T(n) = \sum_{i=1}^{n/2} \sum_{j=i}^{n-i} \sum_{k=1}^{j} 1$

   (b) Simplify the summation. Show your work.

   i. $T(n) = \sum_{i=1}^{n/2} \sum_{j=i}^{n-i} \sum_{k=1}^{j} 1$ From part (a)

   ii. $T(n) = \sum_{i=1}^{n/2} \sum_{j=i}^{n-i} j$ Evaluate inner-most sum

   iii. $T(n) = \sum_{i=1}^{n/2} \sum_{m=1}^{n-2i+1} m + i - 1$ Index shift

   iv. $T(n) = \sum_{i=1}^{n/2} (\sum_{m=1}^{n-2i+1} m + \sum_{m=1}^{n-2i+1} i - 1)$

   v. $T(n) = \sum_{i=1}^{n/2} (\frac{(n-2i+1)(n-2i+2)}{2} + \sum_{m=1}^{n-2i+1} i - 1)$

   vi.

4. Prove the following identities on logarithms:

   (a) $log_a(xy) = log_a(x) + log_a(y)$

i. $m = log_a(x) \rightarrow x = a^m$
ii. $n = log_a(y) \rightarrow y = a^n$
iii. $xy = a^m a^n = a^{m+n}$
iv. $log_a(xy) = log_a(a^{m+n})$
v. $log_a(xy) = (m+n)log_a(a)$
vi. $log_a(xy) = log_a(x) + log_a(y)$

(b) $log_a(x^y) = y\ log_a(x)$

i. Let $m = log_a(x)$
ii. Using the definition of logarithms, we can rewrite this as $x = a^m$
iii. Now, x^y can be rewritten as $(a^m)^y$
iv. We can rewrite this as $log_a(x^y) = ym$
v. Finally, substitute back in our value for m, $log_a(x^y) = y\ log_a(x)$

(c) $log_a(x) = \frac{log_b(x)}{log_b(a)}$

i. $log_a(x) = y \rightarrow a^y = x$
ii. $log_b(a^y) = log_b(x)$
iii. $y\ log_b(a) = log_b(x)$
iv. $y = \frac{log_b(x)}{log_b(a)}$
v. $log_a(x) = \frac{log_b(x)}{log_b(a)}$

(d) $x^{log_b(y)} = y^{log_b(x)}$

i. $x = b^{log_b(x)}$
ii. $y = b^{log_b(y)}$
iii. $x^{log_b(y)} = b^{log_b(x)log_b(y)}$
iv. $y^{log_b(x)} = b^{log_b(x)log_b(y)}$
v. $x^{log_b(y)} = y^{log_b(x)}$

5. You are given 10 bags of gold coins. Nine bags contain coins that each weigh 10 grams. One bag contains all false coins that weigh one gram less. You must identify this bag in just one weighing. You have a digital balance that reports the weight of what is placed on it.

   Begin by labeling your bags 1 through 10. From the first bag, remove one coin. From the second bag, two, etc. Place all of these coins on the scale. Because all of the legitimate bags weigh multiples of 10 grams, we can use the ones digit of the weight to determine which bag the faulty coins came from. For example, if Bag 3 had the counterfeit coins, the weight would end in a 7 because $3 \times 9 = \mathbf{27}$. This can be repeated for every number from 1 to 10, allowing you to determine which bag has the counterfeit coins.