Homework #1 Solution

1. (5 points)

```
Line_num
0       PolyEval(A,x)                       |   |   |     |
1         value = a0                        |   |   |  1  |
2         for i=1 to n-1                    |   |   } n-1 |
3            term = A[i]                    |   | 1 }     |
4            for j=1 to i                   |   } i }     |
5               term = term * x             | 1 }   }     |
6         value = value + term              |   |   |  1  |
7        return value                       |   |   |     |
```

Starting at the innermost loop and working out, we see that the *j* loop has a cost of 1 and executes *i* times on each iteration of the *i* loop. At the same level as the *j* loop is line 3, and we take the max, which is *i* . We see that the *i* loop executes *n-1* times. At the same level as the *i* loop, we see that lines 1 and 6 each have a cost of 1, but again we take the max, giving us :

```
           n-1
T(n) =   Sum(i) =  n( n-1) /2
          i = 1
```

the number of iterations is unaffected by the selected values of coefficients or of x, so the worst-case and best-case running times are the same, and we can say that T(n) = Theta(n^2).

2. **Chapter 1, #14a & b (3 points)**

```
 k                  k    k
Sum (i * 2^i) =  Sum( Sum(2^j) )
i=0               i=1  j=i

                    k
               =  Sum(2^(k+1) - 2^i)
                  i=1
                                k
               =  k*2^(k+1) -  Sum(2^i)
                               i=1

               =  k*2^(k+1) -  (2^(k+1) - 2)

               =  (k-1)k*2^(k+1) + 2
```

14b.

```
 k                  k    k
Sum (i * 2^-i) =  Sum( Sum(2^-j) )
i=0               i=1  j=i

                    k
               =  Sum(2^-i - 2-(k+1)) / (1 - 2^-1)
                  i=1
```

```
                          k
       = -k*2^(-k) + Sum(2^(-i+1))
           k         i=1


       = -k*2^-k + 2 - 2^(-k+1)

       = 2 - (k+2)*2^-k
```

Chapter 1, #16 (3 points)

log_e n = log_2 n * log_e 2, so just multiply the binary log by ln 2 = 0.693147

Chapter 1, #17 (3 points)

17a. Their product is 1 because ....

```
b^(log_b a log_a b) =  (b^(log_b a))^log_a b

                    =  a ^(log_a b)

                    = b
```

17b. log_a x = log_b x * log_a b < log_b x , since log_a b < 1 if a > b

17c. Since n ≥ 1 and the logarithmic functions are monotone increasing,

log_b(n+c) ≤ log_b(nc) = log_b n + log_b c

so the stated inequality holds with d = log_b c.

Chapter 1, #22 (3 points)

22a. False, by part(1) of the Growth Rates Theorem, with $\alpha = 2.5$ and $\beta = 2$.
22b. True, since lg(n^3) ∈ O(log n) ⊆ O(n log n)
22c. True by the Growth Rates and Big-O Theorems, since √(n) ∈ O(√(n)), lg √(n) = 1/2 lg n ∈ O(n ^ (1/2)) = O(√(n)), and hence by multiplying the left and right sides √(n) lg √(n) ∈ O(√(n) √(n)) = O(n).

Chapter 1, #23 (3 points)

23a. True. For any n ≥ 1, 2/n + 4/n^2 ≤ 2/n + 4/n = 6/n ∈ O(1/n).
23b. True, since logarithms to different bases differ only by a constant factor.
23c. True, since logarithms of different powers of n differ only by a constant factor.
23d. False, since log n ∉ √(log_2 n). (If it were true that log n ∈ √(log_2 n), then log n / √(log_2 n) would be bounded by some constant *c* , but taking the base of log n to be 2 this ratio is the monotone increasing function √(log_2 n).)
23e. True, since for n > 26 we have min(700, n^2) < 700 ∈ O(1).

Chapter 1, #36 (6 points)

Using the Divide and Conquer Recurrence Theorem in the text ...

a. a = 3, b = 2, and e=1, so T(n) ∈ O(n^log_2 3).
b. a = 3, b = 2, and e=2, so T(n) ∈ O(n^2).
c. a = 8, b = 2, and e=3, so T(n) ∈ O(n^3 log n).
d. a = 4, b = 3, and e=1, so T(n) ∈ O(n^ log_3 4).

```
e.  a = 4, b = 3, and e=2, so T(n) ∈ O(n^2).
f.  a = 9, b = 3, and e=2, so T(n) ∈ O(n^2 log n).

Using our version of the "Master Method":

a.  a = 3, b = 2, and p=1, k=0, so Case I, and
        T(n) ∈ O(n^log_2 3).
b.  a = 3, b = 2, and p=2, k=0, so Case III, and
        T(n) ∈ O(n^2).
c.  a = 8, b = 2, and p=3, k=0, so Case II, and
        T(n) ∈ O(n^3 log n).
d.  a = 4, b = 3, and p=1, k=0, so Case I, and
        T(n) ∈ O(n^ log_3 4).
e.  a = 4, b = 3, and p=2, k=0, so Case III, and
        T(n) ∈ O(n^2).
f.  a = 9, b = 3, and p=2, k=0, so Case II, and
        T(n) ∈ O(n^2 log n).
```

Chapter 1, #47 (3 points)

```
        36 possibilities (6 x 6)

        1, 1     1, 2    1, 3    1, 4    1, 5    1, 6
        2, 1     2, 2    2, 3    2, 4    2, 5    2, 6
        3, 1     3, 2    3, 3    3, 4    3, 5    3, 6
        4, 1     4, 2    4, 3    4, 4    4, 5    4, 6
        5, 1     5, 2    5, 3    5, 4    5, 5    5, 6
        6, 1     6, 2    6, 3    6, 4    6, 5    6, 6

        Sums of those 36 possibilities:

        2       3       4       5       6       7
        3       4       5       6       7       8
        4       5       6       7       8       9
        5       6       7       8       9       10
        6       7       8       9       10      11
        7       8       9       10      11      12

        sum     count   prob
         2      1       1/36
         3      2       2/36
         4      3       3/36
         5      4       4/36
         6      5       5/36
         7      6       6/36
         8      5       5/36
         9      4       4/36
        10      3       3/36
        11      2       2/36
        12      1       1/36
```

3. **Chapter 2, #15 (3 points)**

```
The outer loop is executed n times; on the ith iteration, the
inner loop iterates i^2 times (since x must be reduced from
i to 0 by steps of 1/i).  So the whole program takes


           n
```

$$\Theta\left(\sum_{i=i} n * i^2\right) \subseteq \Theta(n^3)$$

Chapter 2, #17 (3 points)

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

$$= c \sum_{i=1}^{n-1} \frac{(n+i+1)(n-1)}{2}$$

$$= \frac{c}{2} \sum_{i=1}^{n-1} (n^2 + n = i^2 - i)$$

$$= \frac{c}{2} \left((n^2 + n)(n-1) - \frac{n(n-1)(2n-1)}{6} - \frac{n(n-1)}{2} \right)$$

$$= \frac{c}{3}\ n(n-1)(n+1) \in \Theta(n^3)$$

Chapter 2, #19 (3 points))

$$T(n) = \sum_{i=1}^{floor(sqrt(n))} \sum_{j=1}^{floor(sqrt(n))} \sum_{k=1}^{j} c = c \sum_{i=1}^{floor(sqrt(n))} \sum_{j=1}^{floor(sqrt(n))} j$$

$$= c \sum_{i=1}^{floor(sqrt(n))} \frac{floor(sqrt(n))*(floor(sqrt(n))+1)}{2}$$

$$= \frac{c}{2}\ floor(sqrt(n))^2 (floor(sqrt(n))+1) \in \Theta(n^{3/2})$$

4. a. a = 8, b = 2, p = 3, k=1, so case II, and
   $T(n) = \Theta(n^3 \log^2 n)$

   b. a = 3, b = 2, p = 1, k=0, so case I, and
   $T(n) = \Theta(n^{\log_2 3})$

   c. a = 3, b = 2, p = 2, k=0, so case III, and
   $T(n) = \Theta(n^2)$

   d. a = 16, b = 2, p = 3, k=1, so case I, and
   $T(n) = \Theta(n^4)$

   e. a = 1, b = 10/9, p = 1, k=0, so case III, and
   $T(n) = \Theta(n)$