Text

Description automatically generated

Text

Description automatically generatedText

Description automatically generatedText

Description automatically generated

Text

Description automatically generated

**Problem 3:** Exercise 1.4.6: Give the order of growth (Big-O notation) as a function of n of the running times of each of the following code fragments.  Please justify your reasoning.

1. int sum = 0;  
   for (int k = n; k > 0; k /= 2)  
     for (int i = 0; i < k; i++)  
       sum++;

k = 5; k = 2; 3nlg(n)

sum = 5; sum = 7;

The growth order for this code fragment is 3nlg(n). The reason I say this is because the function will increment by this much between each step. I figured this out mathematically by entering numbers and finding the values that they grew to.

1. int sum = 0;  
   for (int i = 1; i < n; i \*= 2)  
     for (int j = 0; j < i; j++)  
       sum++;

i = 1; i = 2; i = 3; i = 4; 2n

sum = 1; sum = 3; sum = 6; sum = 10;

The growth order for this code fragment is 2n. I say this because the sum is growing in increments of double what the n value is.

1. int sum = 0;  
   for (int i = 1; i < n; i \*= 2)  
     for (int j = 0; j < n; j++)  
       sum++;

i = 1; i = 2; i = 3; i = 4; i = 5; n^2

sum = 5; sum = 10; sum = 15; sum = 20; sum = 25;

The growth order for this code fragment is n^2. I say this because the final value ends up being the square of the value you put in when all of the code has been gone through.

**Problem 4:** Creative Problem 1.4.24: *Throwing eggs from a building!*  Suppose that you have an n-story building and plenty of eggs.  Suppose also that an egg is broken if it is thrown from floor F or higher and intact otherwise.  First, devise a strategy to determine the value of F that uses O(lg n) throws (and so breaks only O(lg n) eggs).  Then, find a way to reduce the number of eggs broken to O(lg F).  *Note: lg n is shorthand for log2 n.*

To find the value of F I would use a binary search method. I would take the max height and the minimum height and I would find the point directly in the middle of them. I would then drop the egg from there. If it breaks, I would move down to halfway between the midpoint and the ground. Otherwise, I would move up into the middle between the midpoint and the max height. Then after each egg drop I would move either up or down depending on if the egg breaks or not and eventually I will find the exact height at which F is defined. The time complexity of this method is lg(n).

If you were to use an insertion sort, you would be able to find the value of F faster. If you were to start at a certain height and increment the same amount every time, dropping at egg at each of these stops. You would then do a bucket sort in the smaller range and your time complexity will be lower.