1) Given a function $f(n)$, fill in each blank with the correct value of $x \in \{1, lg(n), n, nlg(n), n^2, 2^n, n^n\}$ such that the resulting asymptotic bound is as tight as possible, or write N/A if not possible.

$f(n)$	$o(x)$	$O(x)$	$\omega(x)$	$\Omega(x)$	$\Theta(x)$
$4n^2 + n! - 7lg(n) + 1$					
$3^{lg(n)} + 9n$					
$6lg^4(n) + nlg(n)$					

2) Find the expected value of D where X and B are indicator random variables.

$$D = (5!)(X + 1) + log\left(32^B\right)$$

$$Pr\{X \text{ is } 1\} = \frac{1}{10},\ Pr\{B \text{ is } 1\} = \frac{1}{5}$$

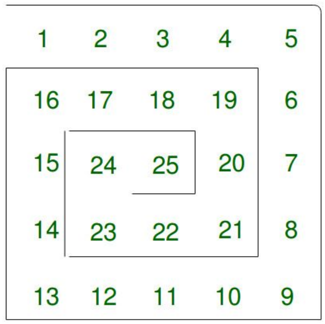
3) Prove that in any full binary tree on n nodes, the number of leaves is exactly one more than the number of internal nodes using mathematical induction.

4) Suppose that we have numbers between 1 and 1000 in a binary search tree, and we want to search for the number 363. Which of the following sequences could not be the sequence of nodes examined?

a. 2, 252, 401, 398, 330, 344, 397, 363
b. 924, 220, 911, 244, 898, 258, 362, 363
c. 925, 202, 911, 240, 912, 245, 363
d. 2, 399, 387, 219, 266, 382, 381, 278, 363
e. 935, 278, 347, 621, 299, 392, 358, 363

5) Professor Ryuto hypothesizes that he can obtain substantial performance gains over a regularly chained hash table by modifying the chaining scheme such that each list is maintained in sorted order. How does this modification affect the running time for successful searches, unsuccessful searches, insertions, and deletions?

6) Write a recursive algorithm that prints out the values in an $n \times n$ 2D matrix in a spiral pattern. Like in the following example:



7) Given the following algorithm, use the master method to find the runtime of the algorithm within a polylog factor of the exact runtime. Then create a recursion tree in order to create a more accurate guess of what the exact runtime should be. Finally use the substitution method to prove that your guess is correct.

$Alg(A, n)$

if $n \leq 3$

    return array

for **i = 0** to n

    $A[i] = A[n - i]$

return $Alg(A[1 \ldots \frac{n}{3}], \frac{n}{3}) \cap Alg(A[\frac{n}{3} + 1 \ldots \frac{2n}{3}], \frac{n}{3}) \cap Alg(A[\frac{2n}{3} + 1 \ldots n], \frac{n}{3})$

8) Using a loop invariant, prove that the Merge function of merge sort is correct. Make sure that your loop invariant fulfills the three necessary properties.

```
// A is an array and p, q, and r are indices into the array such that p <= q < r
Merge(A, p, q, r)
    n1 = q - p + 1
    n2 = r - q
    let L[1.. n1 + 1] and R[1.. n2 + 1] be new arrays
    for i = 1 to n1
        L[i] = A[p + i - 1]
    for j = 1 to n2
        R[j] = A[q + j]
    L[n1 + 1] = infinity
    R[n2 + 1] = infinity
    i = 1
    j = 1
    for k = p to r
        if L[i] <= R[j]
            A[k] = L[i]
            i = i + 1
        else A[k] = R[j]
            j = j + 1
```