1.
a.
<= (2n^2 + 4n^2)/4n = 1.5n, n_0 = 1, c=1.5, O(n)
>= 2n^2/4n = 1/2n, n_0 = 1, c=1/2, O(n)

b.
<= n^2 + 5n^2, n_0=1, c=6, O(n^2)
>= 5n^2, n_0=1, c=5, O(n^2)

c.
<= nlgn, n_0=1, c=1, O(nlgn)
>= ***This was only meant to be a Big-O question***.

2.

The outer loop will run in O(lg n), the inner loop will never run.

3.

Quicksort, merge sort, heapify, binary search

4.

```
if( !root ) return false;
if( root-> empID == key ) { name = root->name; salary = root->salary; return true; }
if( root->empID > key ) return findEmployee( root->left, key, name, salary );
else return findEmployee( root->right, key, name, salary );
```

5.

```
if(!root) return NULL;
// Just the root and no children:
if(!last_parent)
{
   delete root;
   return NULL;
}

if( last_parent->right )
{
    root->key = last_parent->right->key;
    delete (last_parent->right);
    last_parent->right = NULL;
} else {
    root->key = last_parent->left->key;
    delete (last_parent->left);
    last_parent->left = NULL;
```

```
}
reheapDown(root);
return root;
}
```

There could be some variation here depending on reheapDown... if I were to give a question like this I would provide the implementation in the appendix, or at least a function signature so you'd know the specifics of the implementation works.

6.

15 % 13 = 2
17 % 13 = 4
30 % 13 = 4 ==> 4 + (9 - (30%7)) = 4 + (9-2) = 11
4 % 13 = 4 ==> 4 + (9 - (4%7)) = 4 + 5 = 9
54 % 13 = 2 ==> 2 + (9 - (54 % 7)) = 2 + 4 = 6
6 % 13 = 6 ==> 6 + (9 - (6 % 7)) = 6 + 3 = 9 ==> 6 + 2(9 - (6 % 7)) = 12

7.
**Note that i is set to 1 to start, so BC is when i=1!
I(m): exp = x^i

I(1): exp = x^1 = x by LI, and exp = x by program code
I(k): Assume true for 0 <= k <= m
I(k+1):

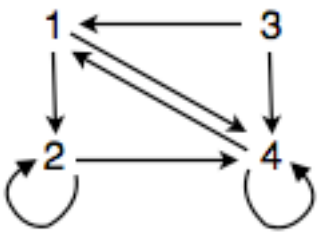
exp_new = exp_old * x
= x^k * x
= x^(k+1) (which matches what is predicted by LI)

8.

a. A, B, F, D, C, E

b. F, D, A, E, C, B

9.



10.

C(13,1)*C(4,3)*C(12,1)*C(4,2)

11.
a. n/2+1

b. n

c. 26^2+1

12. -8, -5, 0, 2, 6, 1, 3, 4, 17 (but drawn as nearly-complete tree)

-5, 2, 0, 4, 6, 1, 3, 17, -8
0, 2, 1, 4, 6, 17, 3, -5, -8
1, 2, 3, 4, 6, 17, 0, -5, -8
2, 4, 3, 17, 6, 1, 0, -5, -8
3, 4, 6, 17, 2, 1, 0, -5, -8
4, 17, 6, 3, 2, 1, 0, -5, -8
6, 17, 4, 3, 2, 1, 0, -5, -8
17, 6, 4, 3, 2, 1, 0, -5, -8

13.

Express as nearly complete tree:
+,-,+,2,*,4,/,empty,empty,3,4,empty,empty,8,2

14.
m

15.
m(m-1)

16.

13	_
7	_		16	21
1	2		8	_		14	_		17	_		22	_

17.

bar(0)
foo(1)
bar(2)

foo(3)
main

foo== isEven
bar== isOdd