2210: Assignment 3 - Allison So 251087238

1. Hash table, size M = 7, hash function h(k) = k mod 7, collisions = separate chaining

0	
1	(1, d_3) → (36, d_4) --|
2	
3	
4	
5	(5, d_1) → (12, d_2) → (26, d_5) --|
6	

2. Hash table, size M = 7, hash function h(k) = k mod 7, collisions = linear probing

0	(26, d_5) --|
1	(1, d_3) --|
2	(36, d_4) --|
3	
4	
5	(5, d_1) --|
6	(12, d_2) --|

3. Hash table, size M = 7, hash function h(k) = k mod 7, collisions = double hashing
   Secondary hash function h'(k) = 5 - (k mod 5)

0	
1	(12, d_2) --|
2	(26, d_5) --|
3	
4	
5	(5, d_1) --|
6	

(1, d_3) and (36, d_4) cannot be input.

4.

c = 0, x = 0, t = 1, res = 0, ret addr =
c = 5, x = 1, t = 1, m = 0, res = , ret addr = A3
c = 10, x = 2, t = 1, m = 5, res = , ret addr = A1
p = 10, res = , ret addr = OS

Execution Stack

5.

```
Algorithm find(r,k,v)
In: Root r of a tree, value k ≥ 0, and value v
Out: Number of nodes at level k storing the value v
        count ← 0
        level ← 0
        if r.isLeaf() then
                if k = level and r.getValue() = v then
                        count = count + 1
                return count
        else
                for each child c of r do
                        level+1
                        if c.getValue() = v and level = k then
                                count+=1
                        else
                                count+=find(c, level-1, v)
        return count
```

6.

Algorithm algo(r)
In: Root r of a proper binary tree storing n nodes
if r is a leaf then return n + func(n)
else {
v ←func(n)
v ← v + algo(r.leftChild()) + algo(r.rightChild())
return v
}
	There are $c_2 + c_1$logn operations performed per call in the first if statement.
	
	Otherwise $c_1$logn operations are performed for func(n), and f((n-1)/2) primitive operations are called in each of the two recursive calls as only half the children are being examined per call.
	
	Therefore the number of primitive operations performed by the algorithm is:
	$f(0) = c_2 + c_1 \log n$
	$f(n) = c_1 \log n + f((n-1)/2) + c_3$ for $n > 0$
	$f(n) = c_1 + \log_2(n+1) + c_3$
	Therefore, the order of this function would be O(log n).