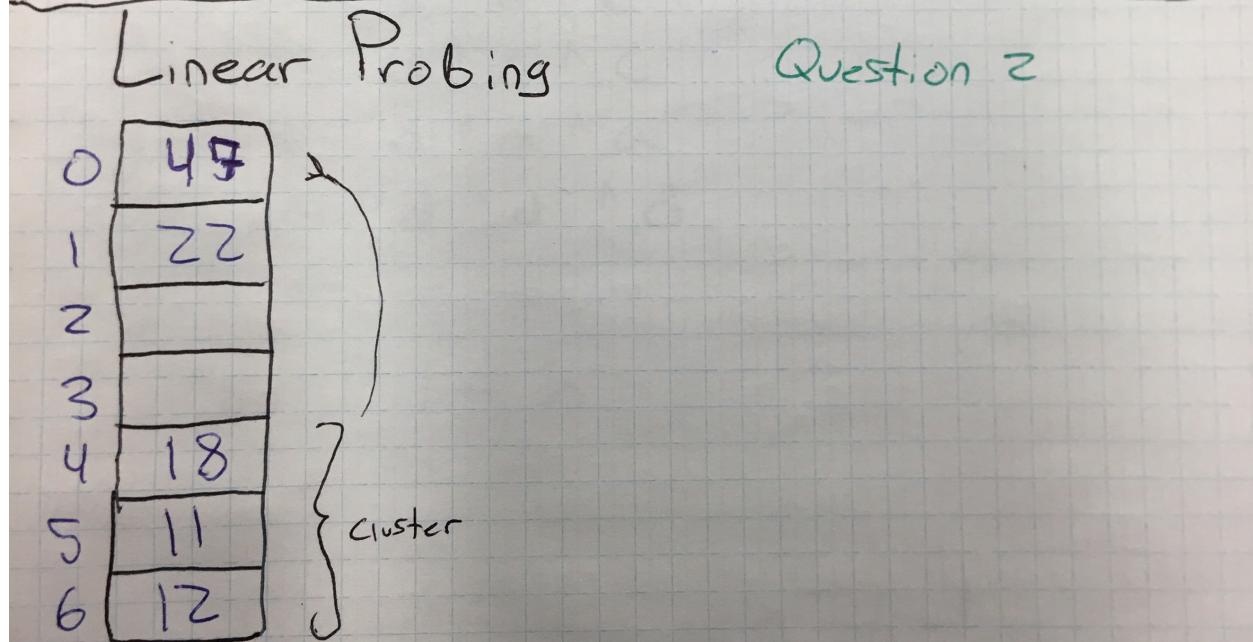
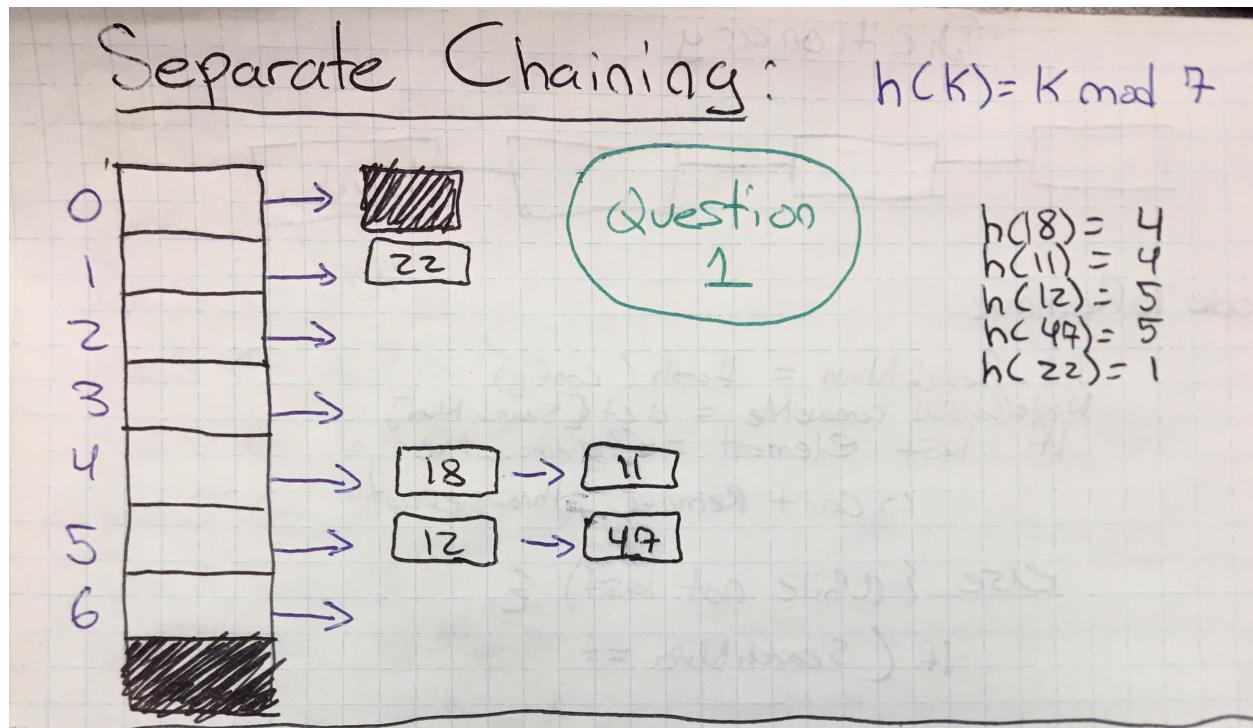


Firas Aboushamalah  
250-920-750  
Assignment #3

## Question 1 & 2



### Question 3:

Double Hashing:	$h(K) = K \bmod 7$	$h'K = 5 - (K \bmod 5)$
	$h(18) = 4$	
	$h(11) = 4$	$h'K = 5 - (11 \bmod 5)$ = $5 - 1 = 4$
	$h(12) = 5$	
	$h(47) = 5$	$h'K = 5 - (47 \bmod 5)$ = $5 - 2 = 3$
	$h(22) = 1$	$h'K = 5 - (22 \bmod 5)$ = $5 - 2 = 3$
0 47		
1 11		
2		
3 22		
4 18		
5 12		
6		

### Question 4:

$$\begin{aligned}
 F(0) &= C_1 \\
 F(n) &= f(n-1) + C_2n + C_3 \text{ for } n > 0 \\
 &= (f(n-2) + C_2(n-1) + C_3) + C_2n + C_3 \\
 &= (f(n-3) + C_2(n-2) + C_2(n-1) + 2C_3) + C_2n + C_3 \\
 &= (f(n-4) + C_2(n-3) + C_2(n-2) + C_2(n-1) + 3C_3) + C_2n + C_3 \\
 &= f(n-k) + C_2(n-k+1) + \dots + C_2n + C_3 \\
 \text{When } n-k=0, n=k \\
 f(k) &= f(n-k) + C_2(n-k+1) + \dots + C_2n + C_3 \\
 &= f(0) + C_2 + \dots + C_2 + C_3 \\
 &= f(0) + \sum_{k=1}^n C_2k + C_3 \\
 \text{Since } f(0)=C_1 & \\
 &= C_1 + \sum_{k=1}^n C_2k + C_3 \\
 &= C_1 + C_2 \sum_{k=1}^n k + C_3 \\
 f(n) &= C_1 + C_3 + C_2 \left( \frac{n(n+1)}{2} \right) \\
 \therefore f(n) &= O(n^2)
 \end{aligned}$$

## Question 5i:

```
Algorithm maxValue(r)
In: Root r of tree
Out: Largest int value stored in node of tree.

IF r.isLeaf then return r;
else {
    max ← 0;
    For each Child c of r do
        Preorder(r);
        IF c.value > max then {
            max ← c.value
        }
    }
    return max;
```

## Question 5ii:

```
C1 IF r.isLeaf then return r;
else {

C2     max ← 0;

    For each Child c of r do
        Preorder(r);
        IF c.value > max then {
            max ← c.value
        }
    }

C4     return max;
```

$$\begin{aligned}f(0) &= C_1 \\F(n) &= C_2 + C_3n + C_4 \\&= O(n)\end{aligned}$$

Add up the constants for size n = 0 and the algorithm ends at first line so  $f(0) = c_1$

For size n, the algorithm has 2 constants and a 3 constant multiplied by n iterations via "preorder(r)"  
Therefore, the time complexity of this algorithm is  $O(n)$  for  $n > 0$