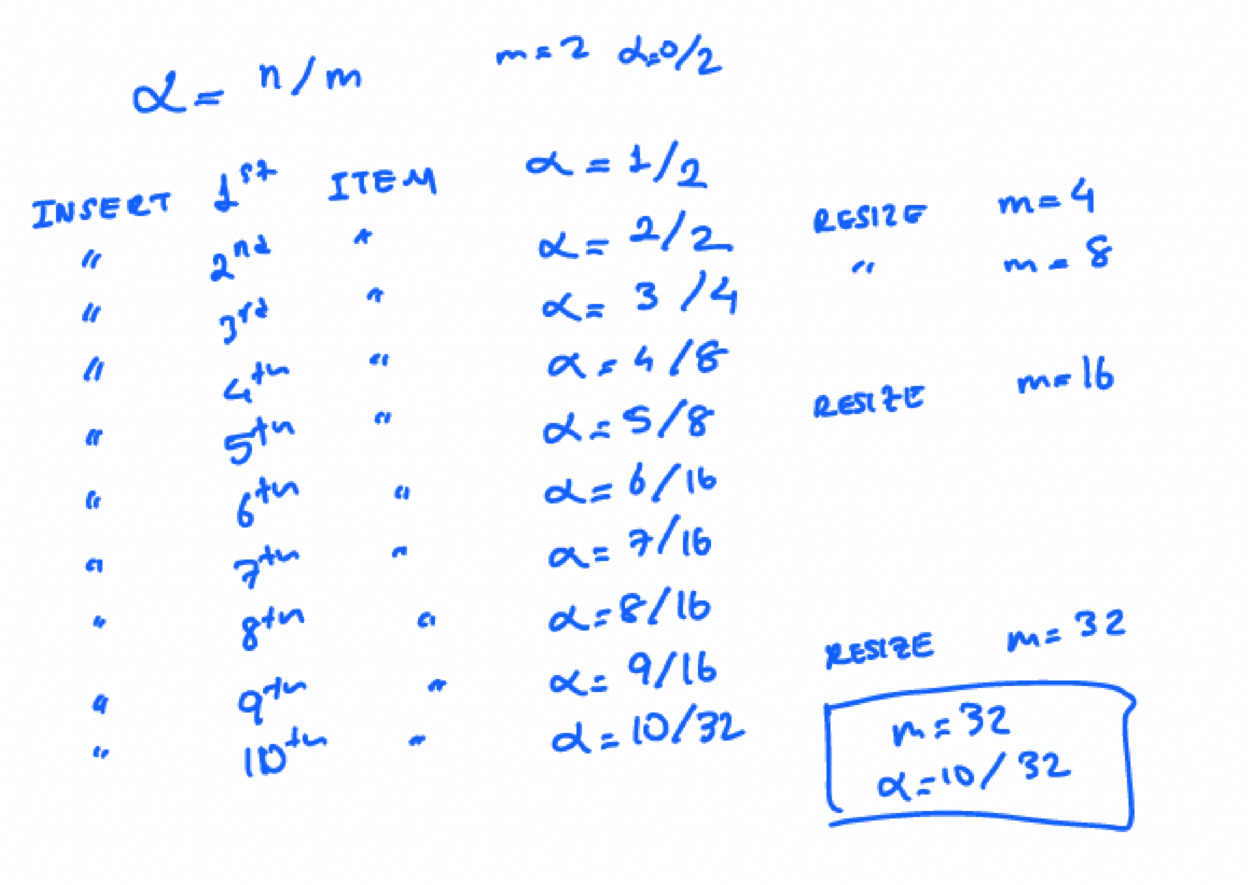
CS 300 Data Structures
Problem Set # 20 – Hashing

1. Let H be a hash-table where collisions are handled by separate chaining and where re-hashing is used each time the load factor (the number of items in the table divided by the size of the table) exceeds ½. We assume that the initial size of H is 2 and that rehashing doubles the size of the table. After inserting 10 items with different keys, what is the size of the hash-table H?

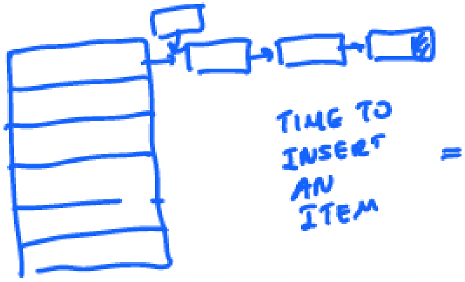


$\alpha = n/m$ $\quad\quad$ $m=2$ $\alpha=0/2$

INSERT	1st	ITEM	$\alpha = 1/2$	RESIZE	$m=4$
"	2nd	"	$\alpha = 2/2$	"	$m=8$
"	3rd	"	$\alpha = 3/4$		
"	4th	"	$\alpha = 4/8$	RESIZE	$m=16$
"	5th	"	$\alpha = 5/8$		
"	6th	"	$\alpha = 6/16$		
"	7th	"	$\alpha = 7/16$		
"	8th	"	$\alpha = 8/16$	RESIZE	$m=32$
"	9th	"	$\alpha = 9/16$		
"	10th	"	$\alpha = 10/32$		

$$\boxed{\begin{array}{l} m=32 \\ \alpha=10/32 \end{array}}$$

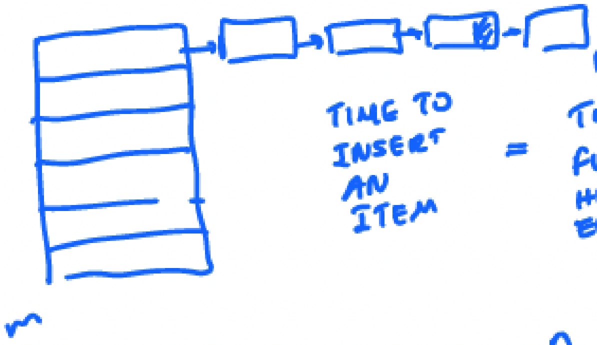
2. Consider an initially empty hash table of size M and hash function h(x) = x mod M. In the worst case, what is the time complexity to insert n different keys into the table if separate chaining is used to resolve collisions (without re-hashing)? Suppose that each entry of the table stores an unordered linked list. When adding a new element to an unordered linked list, such an element is inserted at the beginning of the list.



$$\text{TIME TO INSERT AN ITEM} = \overset{O(1)}{\text{TIME TO FIND HASH ENTRY}} + \overset{O(1)}{\text{TIME TO INSERT}} > \underline{\underline{O(1)}}$$

$$n \cdot O(1) \Rightarrow O(n)$$

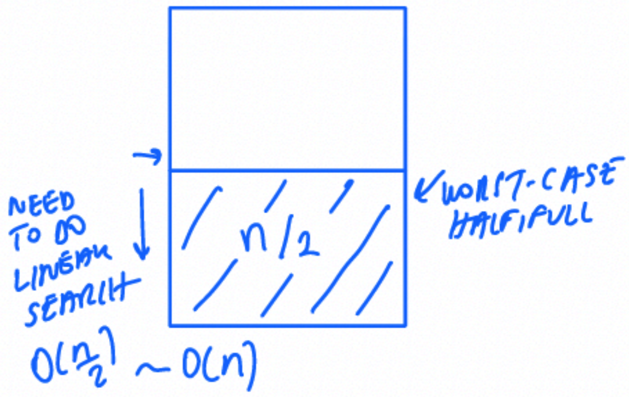
3. What is the answer for question 3 if the linked lists are ordered?



$$\text{TIME TO INSERT AN ITEM} = \overset{O(1)}{\text{TIME TO FIND HASH ENTRY}} + \overset{\underline{O(n)}}{\text{TIME TO INSERT}}$$

$$n \cdot O(n) = \underline{\underline{O(n^2)}}$$

4. Consider the same hash table as above, but assume now that collisions are resolved using linear probing, and n<=M/2. In the worst case, what is the time complexity to insert n keys into the table?



1 INSERT $O(n)$

n INSERTS $O(n^2)$