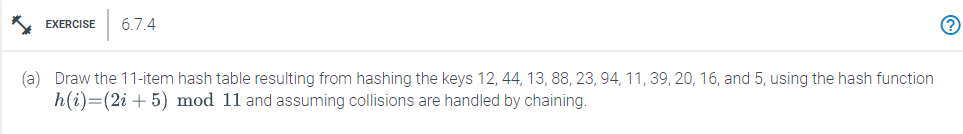
# Michael Chillemi

# 06/23/2023

# CS 590 - Algorithms

# M5.B1: Module 5 Hash Tables Reinforcement Exercises

Problem 6.7.4



Answer:

Given Hash Function: h(i) = (2i + 5) mod 11

Given Hash Keys: {12,44,13,88,23,94,11,39,20,16,5}

Assuming that the collisions are handled by chaining then the 11 item hash table should look like:

H[0] = NULL

H[1] -> 20

H[2] = NULL

H[3] = NULL

H[4]- > 16, 5

H[5] -> 44,88,11

H[6] -> 94,39

H[7] -> 12,23

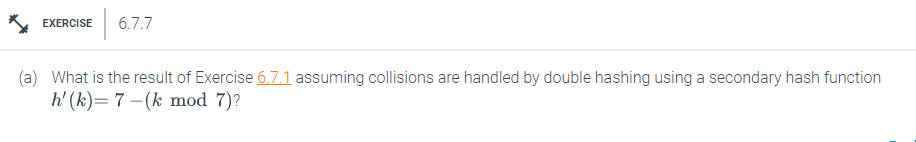
H[8] = NULL

H[9]->13

H[10] = NULL

This approach resolves collisions by chaining collisions. In place of a key, the ith entry in the hash table H[0 / n - 1], where i = 0,....n-1, contains a pointer. The linked list's keys are all defined by the formula h(k) = i. Now h(12) = (2\*12 + 5) mod 11 = 29 %11 = 7 12 is therefore at position 7h(5) = (2\*5 + 5) mod 11 = 15 %11 = 4. Location 4 then changes to location 5, and so forth.

Problem 6.7.7



Answer:

We can observe all the indices where there are many elements from the hash function result shown above. We must use double hashing to determine these positions. Let's continue through the list in the following order: 12, 44, 13, 88, 23, 94, 11, 39, 20, and 16. Since element 12 was the first element to hash into index 8, there is no need to move it. This it will stay in place. Similar to element 12, element 44 is the first to hash into index 5. Element 13 will remain at its index, exactly like elements 12 and 44 did. Since element 88 is at index 5, where element 44 already resides, we will use the double hash function, which is 7 - (88 mod 7) = 3. By adding 3 to its existing index, 5 + 3 = 8, we shift 88. Since index 8 is already taken, we advance 88 by three positions. (8 + 3) % 11 =0. Index 0 is once more filled, therefore 0 plus 3 equals 3. Index 3 is currently empty, thus we move 88 to position 3. Since index 8 is already taken by element 23, we apply the double hashing function, and we obtain 5. Now move 5 positions to its present position, which is 8 + 5% of 11. To keep it within the 0 to 10 index, we are taking the mod of 11. New position is therefore 2. Because it is empty, element 94 will continue to be at position 1. Since there is already an element at element 11's present position, its new position will be index 6, therefore using the double hash function, we get 7-11%7=3. We now advance three positions till we find a new slot that is open. Therefore, the initial movement is 5+3%11=8, followed by 8+3%11=0, 0+3%11, and finally 3+3%11=6. Since slot 6 is vacant, element 11 is moved there. Due to the fact that element 39's present place is already taken, double hashing will be used. We get 7-(39%7)=3. So we shift three positions. we get 1+3%11 = 4. It would go to position 4 since it is vacant. Element 20 will remain in the same place. Element 16 will stay in the same place. The equivalent new location for element 5 would be 7.1

Final Hash Table:

H[0] = 13

H[1] = 94

H[2] = 23

H[3] = 88

H[4] = 39

H[5] = 44

H[6] = 11

H[7] = 5

H[8] = 12

H[9] =16

H[10] = 20