**CS 5012: Foundations of Computer Science**

**3.6 Homework: Hashing Conflict Resolution**

**H. Diana McSpadden (hdm5s)**

**2/19/2021**

## EXERCISE 1: Hashing and Chaining with String Keys

| | | | | |
|---|---|---|---|---|
| 0 | | okra | potato | |
| 1 | | onion | carrot | |
| 2 | | | | |
| 3 | | | | |
| 4 | | cabbage | | |
| 5 | | | | |
| 6 | | mushroom | | |
| 7 | | Salt | | |
| 8 | | | | |
| 9 | | cucumber | | |
| 10 | | tomato | mellon | olive |
| 11 | | banana | | |
| 12 | | orange | | |

## EXERCISE 2: Hashing and Linear Probing

### Resulting Hash Table

| Index | Status | Value |
|---|---|---|
| 0 | O | 26 |
| 1 | E | |
| 2 | E | |
| 3 | E | |
| 4 | E | |
| 5 | O | 18 |
| 6 | E | |
| 7 | E | |
| 8 | O | 47 |
| 9 | O | 35 |
| 10 | O | 9 |
| 11 | E | |
| 12 | O | 64 |
| 13 | E | |

### Operation Table

| Operation | Index or Probe Sequence | Comment |
|---|---|---|
| Insert(18) | h0(18) = 18%13 = 5 | Success |
| Insert(26) | h0(26) = 26%13 = 0 | Success |
| Insert(35) | h0(35) = 35%13 = 9 | Success |
| Insert(9) | h0(9) = 9%13 = 9 | Collision-Success with probing |
| Find(15) | h0(15) = 15%13 = 2 | Fail |
| Find(48) | h0(48) = 48%13 = 9 | Collison, Collison, Fail |
| Find(9) | h0(9) = 9%13 = 9 | Collision-Success with probing |
| Insert(64) | h0(64) = 64%13 = 12 | Success |
| Insert(47) | h0(47) = 47%13 = 8 | Success |
| Find(35) | h0(35) = 35%13 = 9 | Success |

### EXERCISE 3: Additional questions

## Q1) Name one advantage of Chaining over Linear Probing.

Advantages include that the table will never fill up and that the size of the primary hashing table is known based on the hashing function.

## Q2) Name one disadvantage of Chaining that isn’t a problem in Linear Probing.

The table can become unbalanced, with long chains for some nodes, and with long chains come inefficiencies.

## Q3) If using Chaining, how can finding an element in the linked list be made more efficient?

I think a double hashing function that gives the index for the primary hash table, and the double hash would be a good method to improve searching/finding/probing. Alternatively, since the chained list is hopefully short in comparison to n, the chained list could be sorted and then a binary search algorithm could be used; I would only recommend this approach for a read-heavy/write-light scenario.

## Q4) Why does Linear Probing require a three-state (Occupied, Empty, Deleted) “flag” for each cell, but Chaining does not? You may use an example as an illustration to your argument.

If an item was simply deleted, and there was no way to indicate “no value” due to no values vs. deletion, then any matching hash value items “below” the deleted item would be orphaned and would either be unable to be found or require a sequential search of the entire table to guarantee they were no longer in the table.