COP 3530 – Data Structures and Algorithms

Homework-6 (Total 100 points) – Daniel Taylor

1. Given a hash table of size 17, input of {43, 71, 13, 40, 61, 73, 39, 99, 44, 16} and the hash function

h(x) = x % 17, where x is the input value and h(x) are the resulting hash value, show the hash tables resulting from

1. Linear probing

● In the case of a collision, keep going to the next hash table location until find an  
open location  
● If table[i] is occupied, check table[i+1], table[i+2], table[i+3], ...  
● Need 3 states for each hash table location:

empty, occupied, deleted. Use value -1 for dummy table values. (Empty)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input | Result of x % 17 | **Hash Table Index** | **Hash Table Value** | State |
| 43 | **9** | **0** | **-1** | Empty |
| 71 | **3** | **1** | **-1** | Empty |
| 13 | **13** | **2** | **-1** | Empty |
| 40 | **6** | **3** | **71** | Occupied |
| 61 | **10** | **4** | **-1** | Empty |
| 73 | **5** | **5** | **73** | Occupied |
| 39 | **5** | **6** | **40** | Occupied |
| 99 | **14** | **7** | **39** | Occupied |
| 44 | **10** | **8** | **-1** | Empty |
| 16 | **16** | **9** | **43** | Occupied |
|  |  | **10** | **61** | Occupied |
|  |  | **11** | **44** | Occupied |
|  |  | **12** | **-1** | Empty |
|  |  | **13** | **13** | Occupied |
|  |  | **14** | **99** | Occupied |
|  |  | **15** | **-1** | Empty |
|  |  | **16** | **16** | Occupied |

1. Quadratic probing

table[i] is occupied, check table[i+1^2], table[i+2^2 ], table[i+3^2 ]...

Will repeat through table if size limit is reached

**39’s result is at index 5, which is occupied. Check i + (i^2) which is index 6, which is occupied, so check i + (2^2) = i + 4 = so index 9. This is taken by 43, so continue probing and check i + (3^2) = i + 9, so index 14.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input | Result of x % 17 | **Hash Table Index** | **Hash Table Value** | State |
| 43 | **9** | **0** | **-1** | Empty |
| 71 | **3** | **1** | **-1** | Empty |
| 13 | **13** | **2** | **-1** | Empty |
| 40 | **6** | **3** | **71** | Occupied |
| 61 | **10** | **4** | **-1** | Empty |
| 73 | **5** | **5** | **73** | Occupied |
| 39 | **5** | **6** | **40** | Occupied |
| 99 | **14** | **7** | **-1** | Empty |
| 44 | **10** | **8** | **-1** | Empty |
| 16 | **16** | **9** | **43** | Occupied |
|  |  | **10** | **61** | Occupied |
|  |  | **11** | **44** | Occupied |
|  |  | **12** | **-1** | Empty |
|  |  | **13** | **13** | Occupied |
|  |  | **14** | **39** | Occupied |
|  |  | **15** | **99** | Occupied |
|  |  | **16** | **16** | Occupied |

1. Separate chaining

● A hash table that has linked list (a chain) at each location table[i], so that items of

the same hash index are stored here

● Size of the table is dynamic

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Input | Result of x % 17 | **Hash Table Index** | **Hash Table Value** | State |
| 43 | **9** | **0** | **-1** | Empty |
| 71 | **3** | **1** | **-1** | Empty |
| 13 | **13** | **2** | **-1** | Empty |
| 40 | **6** | **3** | **71** | Occupied (Number of items in index: 1) |
| 61 | **10** | **4** | **-1** | Empty |
| 73 | **5** | **5** | **73 -> 39** | Occupied (Number of items in index: 2) |
| 39 | **5** | **6** | **40** | Occupied (Number of items in index: 1) |
| 99 | **14** | **7** | **-1** | Empty |
| 44 | **10** | **8** | **-1** | Empty |
| 16 | **16** | **9** | **43** | Occupied (Number of items in index: 1) |
|  |  | **10** | **61 -> 44** | Occupied (Number of items in index: 2) |
|  |  | **11** | **-1** | Empty |
|  |  | **12** | **-1** | Empty |
|  |  | **13** | **13** | Occupied (Number of items in index: 1) |
|  |  | **14** | **99** | Occupied (Number of items in index: 1) |
|  |  | **15** | **-1** | Empty |
|  |  | **16** | **16** | Occupied (Number of items in index: 1) |

1. Double hashing (using a secondary hash function h’(x) = 7 – x % 7).

If computed index is occupied, do the secondary hash function, add step size to the current index. New index = (Index + i \* Step size) % 17, with i = probe count (number of collisions)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Input | Result of (h)x = x % 17 | Result of h’(x) = 7 – x % 7, if needed | New index, if needed | **Hash Table Index** | **Hash Table Value** | State |
| 43 | **9** | **N/A** | **N/A** | **0** | **-1** | Empty |
| 71 | **3** | **N/A** | **N/A** | **1** | **-1** | Empty |
| 13 | **13** | **N/A** | **N/A** | **2** | **-1** | Empty |
| 40 | **6** | **N/A** | **N/A** | **3** | **71** | Occupied (Number of times for secondary hash: 0) |
| 61 | **10** | **N/A** | **N/A** | **4** | **-1** | Empty |
| 73 | **5** | **N/A** | **N/A** | **5** | **73** | Occupied (Number of times for secondary hash: 0) |
| 39 | **5** | **3** | **8** | **6** | **40** | Occupied (Number of times for secondary hash: 0) |
| 99 | **14** | **N/A** | **N/A** | **7** | **-1** | Empty |
| 44 | **10** | **5** | **15** | **8** | **39** | Occupied (Number of times for secondary hash: 1) |
| 16 | **16** | **N/A** | **N/A** | **9** | **43** | Occupied (Number of times for secondary hash: 0) |
|  |  |  |  | **10** | **61** | Occupied (Number of times for secondary hash: 0) |
|  |  |  |  | **11** | **-1** | Empty |
|  |  |  |  | **12** | **-1** | Empty |
|  |  |  |  | **13** | **13** | Occupied (Number of times for secondary hash: 0) |
|  |  |  |  | **14** | **99** | Occupied (Number of times for secondary hash: 0) |
|  |  |  |  | **15** | **44** | Occupied (Number of times for secondary hash: 1) |
|  |  |  |  | **16** | **16** | Occupied (Number of times for secondary hash: 0) |