William Faunce

Carol Roberts

COS 225

27 April 2018

A10WilliamFaunce

Part I

1.

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 26 | 13 |  | 10 | 45 | 70 | 18 |  | 5 |  |  |  | 25 |

0 1 2 3 4 5 6 7 8 9 10 11 12

2 probes are required to determine value 13 is in the hash table

3 probes are required to determine value 38 is not in the hash table

|  |
| --- |
|  |
|  |
|  |
|  |
|  |

2. a

Table[0] 20 30

Table[1] 6

Table[2]

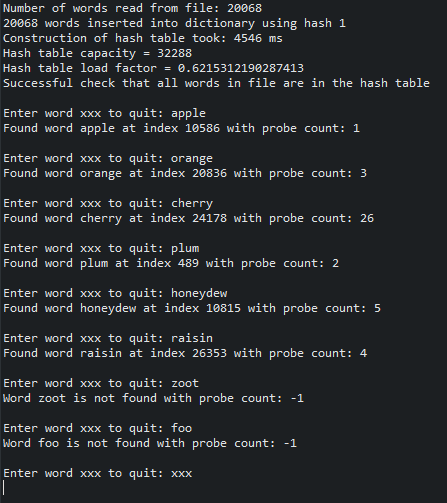
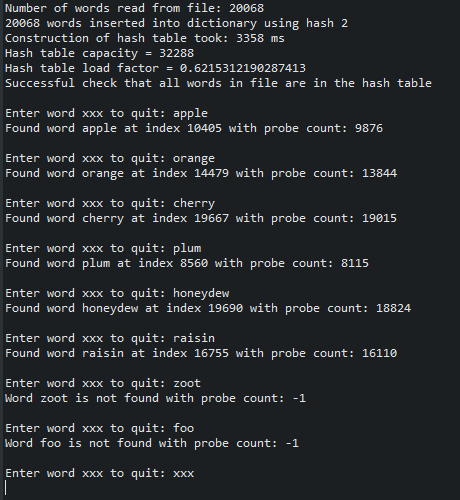
Table[3] 13 8

Table[4] 14

1 probes are required to determine value 13 is in the hash table

2 probes are required to determine value 38 is not in the hash table

Screenshots:



Analysis:

Hash1:

Based on my test runs hash1 can be considered a good hashing function. This is because hash1 uses the line “int seed = 131;” which adds to the “randomness” of the index that the word will get assigned. The value of the seed gets multiplied by the sum and is then added to the total integer value of the word being hashed.

Hash2:

Based on my test runs hash2 can be considered a bad hashing function. In the screenshot you can see that there are thousands of more probes to find a word since it is likely that the words are having collisions with others. This is most likely because in hash2 there is no “seed” that randomizes the hash index a bit.

Also the sum is only declared as an int so its maximum value is limited compared to hash1 which long to declare sum.