

## Practice Exercise #47: Hashing with Double Hashing

[http://www.comp.nus.edu.sg/~cs1020/4\\_misc/practice.html](http://www.comp.nus.edu.sg/~cs1020/4_misc/practice.html)

### Objectives:

- Implementing a hash table
- Implementing double hashing for collision resolution

### Task statement:

You are given the **TableEntry** class for the hash table, which is an array of objects of this class. It contains the following attributes:

- An integer **key**
- A string **text** that represents the data
- A boolean flag **isEmpty** to indicate whether that particular entry is empty or not.

A **Hashing** class that provides a hash table for 31 entries is also given. It uses the following:

- Hash function: **hash(key) = key % 31**
- **Linear probing** to handle collisions

A client program **TestHashingV1.java** is given. It reads in a set of data and insert them into the hash table, using linear probing for collision resolution.

Using **data2.in** (see below) for example, we will illustrate how the code works. The first line contains an integer indicating the number of data sets, followed by the data sets, one on each line.

```
20
2417 BWTXPXJCVGBCZHUTUTIYPDLBCR
807 TMMHEUBZKLQK
4088 LSQYQJ
6813 TS
9784 JRCCWLQRKGFNUIRCCQGOKYV
2260 TEZJQMOZJZYCAWLCKHP
3119 NLPXZN
2818 HGBDVF
9966 ALJZHOKWDNILIOBFEQSIEMSMD
1113 AULGZVNKTCGKTG
8868 UAOPUHVCZHV
2314 HGFCUFSPFMDOJRUHLHZXLE
4466 UQUQLIULWMDHBRHEL
1637 UZHC
1626 XELWRYITJBJECJMYVYXJNINDKB
1946 CFSSL
3755 VQGBACW
8631 YWNRZKZKRKJ
5847 XHCNOVFGVIAQPPJPOIAGIXWF
3055 NTBASQCKNXWLZJSQXTOCYXVMOSF
```

We will trace the addition of the 4 keys highlighted above. After the addition of the first 6 data sets, we have:

Hash table:

```
1: 807, TMMHEUBZKLQK
19: 9784, JRCCWLQRKGZFNUIRCCQGOKYV
24: 6813, TS
27: 4088, LSQYQJ
28: 2260, TEZJQMOZJZYCAWLCKHP
30: 2417, BWTXPXJCVGBCZHYTUTIYPDLBCR
```

In adding key 3119, the key is hashed to 19 ( $3119 \% 31 = 19$ ). As the location is already occupied by key 9784 (also hashed to 19), collision occurs. Using linear probing, the next available location is at 20. Number of collisions is 1.

key 3119 hashed to 19

Number of collisions = 1

Hash table:

```
1: 807, TMMHEUBZKLQK
19: 9784, JRCCWLQRKGZFNUIRCCQGOKYV
20: 3119, NLPXZN
24: 6813, TS
27: 4088, LSQYQJ
28: 2260, TEZJQMOZJZYCAWLCKHP
30: 2417, BWTXPXJCVGBCZHYTUTIYPDLBCR
```

In adding key 2818, the key is hashed to 28 ( $2818 \% 31 = 28$ ). As the location is occupied by key 2260 (also hashed to 28), collision occurs again. Using linear probing, the next available location is at 29. Number of collisions is 1.

key 2818 hashed to 28

Number of collisions = 1

Hash table:

```
1: 807, TMMHEUBZKLQK
19: 9784, JRCCWLQRKGZFNUIRCCQGOKYV
20: 3119, NLPXZN
24: 6813, TS
27: 4088, LSQYQJ
28: 2260, TEZJQMOZJZYCAWLCKHP
29: 2818, HGBDVF
30: 2417, BWTXPXJCVGBCZHYTUTIYPDLBCR
```

In adding key 9966, the key is hashed to 15 ( $9966 \% 31 = 15$ ). As the location is not occupied, there is no collision.

key 9966 hashed to 15

Number of collisions = 0

Hash table:

1: 807, TMMHEUBZKLQK  
**15: 9966, ALJZHOKWDNILIOBFEQSIEMSMD**  
19: 9784, JRCCWLQRKGZFNUIRCCQGOKYV  
20: 3119, NLPXZN  
24: 6813, TS  
27: 4088, LSQYQJ  
28: 2260, TEZJQMOZJZYCAWLCKHP  
29: 2818, HGBDVF  
30: 2417, BWTXPXJCVGBCZHYTUTIYPDLBCR

In adding key 1113, the key is hashed to 28 ( $1113 \% 31 = 28$ ). As the location is occupied by key 2260 (also hashed to 28), collision occurs again. Using linear probing, the next available location is at 0, as locations 29 and 30 are occupied as well. Number of collisions is 3.

key 1113 hashed to 28

Number of collisions = 3

Hash table:

**0: 1113, AULGZVNKTCGKTG**  
1: 807, TMMHEUBZKLQK  
15: 9966, ALJZHOKWDNILIOBFEQSIEMSMD  
19: 9784, JRCCWLQRKGZFNUIRCCQGOKYV  
20: 3119, NLPXZN  
24: 6813, TS  
27: 4088, LSQYQJ  
28: 2260, TEZJQMOZJZYCAWLCKHP  
29: 2818, HGBDVF  
30: 2417, BWTXPXJCVGBCZHYTUTIYPDLBCR

Your task is to add a method **doubleHashing()** in **Hashing.java** to implement double hashing, with the following second hash function:

$$\text{hash}(\text{key}) = 23 - (\text{key} \% 23)$$

and write **TestHashingV3.java** to test it out. You are to submit the new **Hashing.java** and **TestHashingV3.java**.

## Sample Input

10  
1075 UBAKJCPOMCY  
3977 PYNUJNSIIJZTBOUROJJGMBPMCMCLL  
7488 VCTP  
379 PIKYX  
2866 DNAP  
792 PCHAWUDOSPKLHQDLNLAOEGV  
6054 VHQPVZDOJGFSPR  
1830 JAEUUQNPJ  
5112 KUNDTBEFIENXUMDXCFHWYBPKVP  
8551 GZQGZEHJBLUGCSTDGUHUJTIHU

## Sample Output

The output is long. Only partial output is shown below.

:  
key 2866 hashed to 14  
Number of collisions = 0  
Hash table:  
7: 379, PIKYX  
9: 3977, PYNUNJNSIIJZTBOUROJJGMBPMCMCLL  
14: 2866, DNAP  
17: 7488, VCTP  
21: 1075, UBAKJCPOMCY  
  
key 792 hashed to 17  
Number of collisions = 1  
Hash table:  
7: 379, PIKYX  
9: 3977, PYNUNJNSIIJZTBOUROJJGMBPMCMCLL  
14: 2866, DNAP  
17: 7488, VCTP  
21: 1075, UBAKJCPOMCY  
30: 792, PCHAWUDOSPKLHQDLNLAOEGV  
  
:  
Total collisions = 2