10710EECS204001  
Data Structures Homework 6

Due date: 2018/1/1 23:59

Submit to OJ: #12097

Upload code to iLMS

Submission

* Please **1)** submit your code to OJ (OJ: #12097),   
  and **2)** upload the zipped file (source codes) to iLMs.   
  **Both should be done before the due date.**
* Scores will be given based on your OJ results, and the uploaded zipped file (the source codes) should be identical to those submitted to OJ. TAs will examine your uploaded codes.

Description

In this homework, you are asked to implement a hash table for strings. You need to implement the following three functions.

1. **add**

Given a string, insert the string into the hash table according to hash function. If the given string already exists in the hash table, do nothing.

1. **delete**

Delete a given string from the hash table. If the given string does not exist in the hash table, do nothing.

1. **exist**

Check whether a given string exists in the hash table.

**Hash table**

Each entity in the hash table stores strings with a specific key. Please resolve **collisions with chaining.** In this homework, the number of buckets is fixed to **41**. The following figure (Fig. 1)is an example of the hash table.

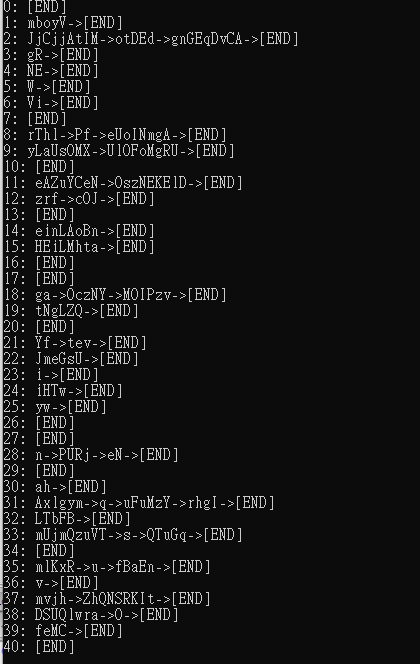


Fig. 1

**Hash function**

The hash function maps an input string to an integer in the range . In this homework, you need to transform each string into a 13331-based (13331進位) integer and modulo 41. Each digit of the given string is defined by its ASCII code (<https://zh.wikipedia.org/wiki/ASCII>).

For example, give the string “abcde”, the corresponding ASCII code is “

97 98 99 100 101”, and the hash value is computed as:

For another example, given the string “AbZz”, the corresponding ASCII code is “65 98 90 122”, and the hash value is thus:

HINT: Computing the value directly would cause an overflow. Please refer to the modulo multiplication and modulo addition.

https://www.khanacademy.org/computing/computer-science/cryptography/modarithmetic/a/modular-multiplication

https://www.khanacademy.org/computing/computer-science/cryptography/modarithmetic/a/modular-addition-and-subtraction

Input

A sequence of commands consisting *“add”, “exist”, “delete”* and *“print”*. Commands *“add”, “exist”* and *“delete”* will be followed by a string. For example, “*add abcd”* means adding the string “abcd” into the hash table, and “*exist bcde*” means checking whether the string “bcde” exists in the hash table. Each string contains at most 10 alphabetical characters (‘a-zA-Z’).

Command *“print”* outputs every element in the hash table. The print function has been defined and implemented in the header file.

output

When command *“exist”* is called, output “exist” if the given string exists in the hash table. Otherwise, output “not exist”. When command *“print”* is called, print every element in the hash table. The format to print is defined in the header file.

Sample Input

add ah

add Vi

add n

add yw

add PURj

add mlKxR

add Yf

add iHTw

add zrf

add JjCjjAtIM

add ga

add einLAoBn

add tev

add eN

add i

add yLaUsOMX

add Axlgym

add OczNY

add u

add mUjmQzuVT

add JmeGsU

add mvjh

add v

add eAZuYCeN

add MOIPzv

add fBaEn

add s

add rThl

add q

add otDEd

add UlOFoMgRU

add HEiLMhta

add QTuGq

add uFuMzY

add OszNEKElD

add feMC

add ZhQNSRKIt

add gnGEqDvCA

add NE

add gR

add cOJ

add DSUQlwra

add mboyV

add LTbFB

add W

add rhgI

add O

add Pf

add tNgLZQ

add eUoINmgA

exist O

delete O

exist O

print

Sample output

exist

not exist

0: [END]

1: mboyV->[END]

2: JjCjjAtIM->otDEd->gnGEqDvCA->[END]

3: gR->[END]

4: NE->[END]

5: W->[END]

6: Vi->[END]

7: [END]

8: rThl->Pf->eUoINmgA->[END]

9: yLaUsOMX->UlOFoMgRU->[END]

10: [END]

11: eAZuYCeN->OszNEKElD->[END]

12: zrf->cOJ->[END]

13: [END]

14: einLAoBn->[END]

15: HEiLMhta->[END]

16: [END]

17: [END]

18: ga->OczNY->MOIPzv->[END]

19: tNgLZQ->[END]

20: [END]

21: Yf->tev->[END]

22: JmeGsU->[END]

23: i->[END]

24: iHTw->[END]

25: yw->[END]

26: [END]

27: [END]

28: n->PURj->eN->[END]

29: [END]

30: ah->[END]

31: Axlgym->q->uFuMzY->rhgI->[END]

32: LTbFB->[END]

33: mUjmQzuVT->s->QTuGq->[END]

34: [END]

35: mlKxR->u->fBaEn->[END]

36: v->[END]

37: mvjh->ZhQNSRKIt->[END]

38: DSUQlwra->[END]

39: feMC->[END]

40: [END]