Suppose we have the following info …

| Locker# | Name |
| --- | --- |
| 2 | YoungChan |
| 25 | Taewhan |
| 10 | HeeChun |
| 49 | Dongyoon |
| 82 | ByungMin |
| ….. | ….. |

… then, we want to retrieve the name, given a locker number.

Now suppose our keys are not so nicely described …

Course Number -> Schedule info

Color -> BMP (bitmap image file)

Vertex -> Set of incident edges

Flight number -> arrival information

URL -> html page

# Dictionary in the context of hashing

*A dictionary* is a structure supporting the following:

void insert(kType & k, dType & d)

void remove(kType & k)

dType find(kType & k)

In hashing, an *associative array* is a dictionary with a particular interface:

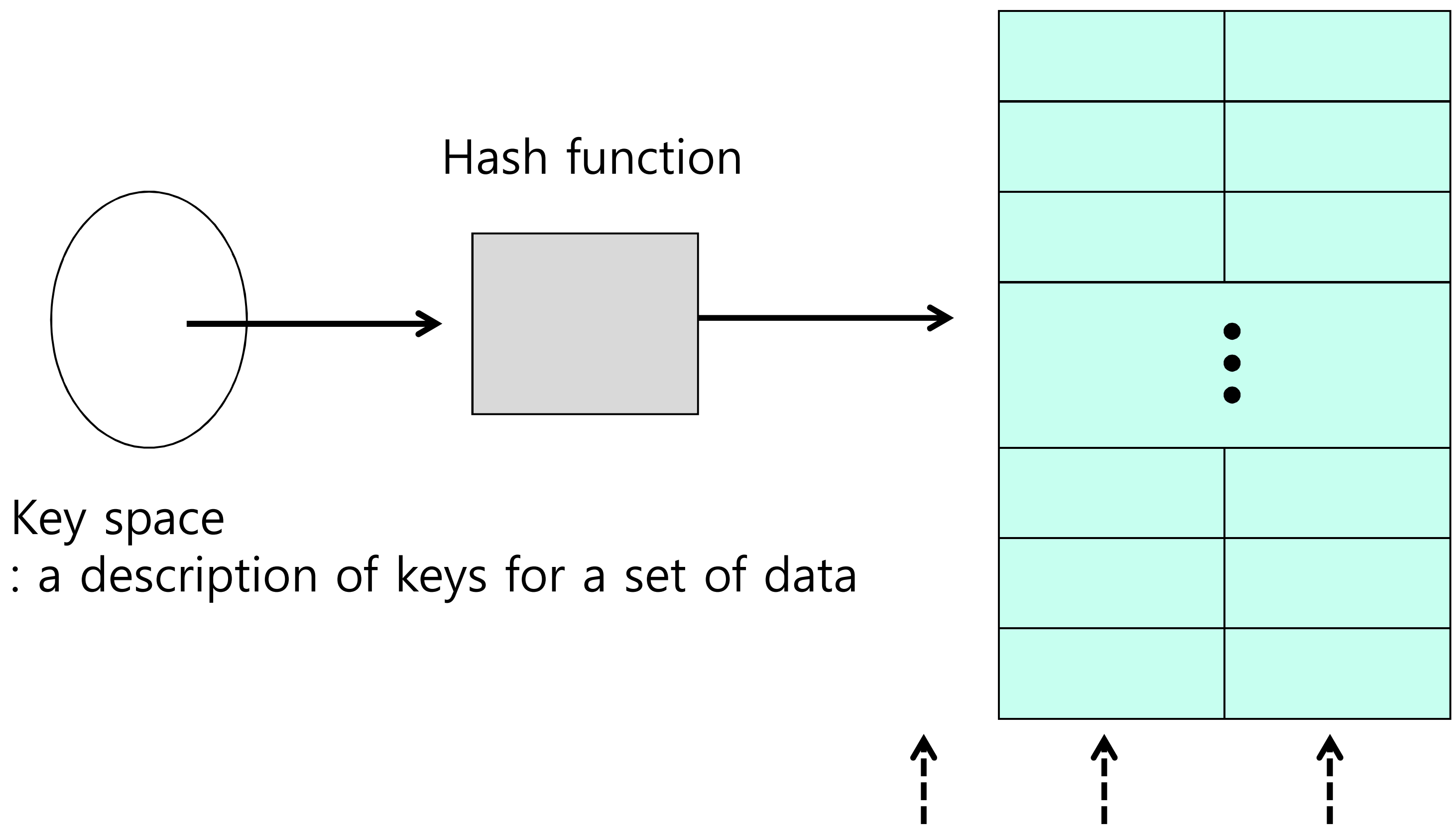
Overloads the [ ] operator for insert and find:

myDictionary[“Youngchan”] = 22;

dType d = myDictionary[“Byungmin”];

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

## Basic Idea: we seek a mapping h(k)



## Perfect hashing

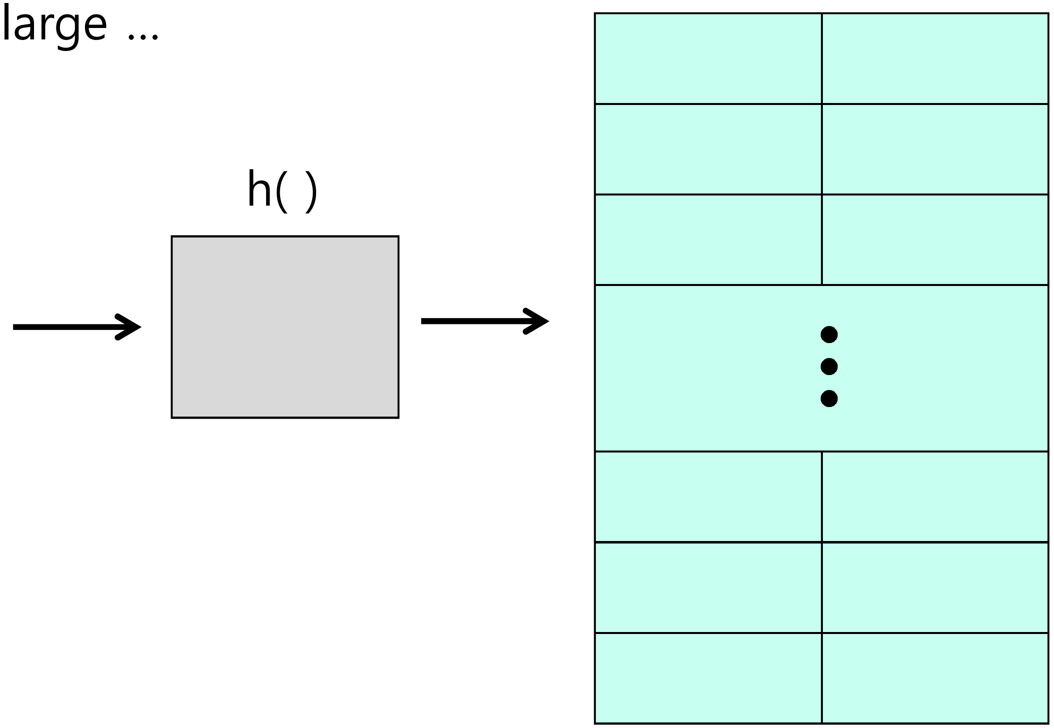
A mapping function is *perfect hashing function* if it satisfies:

- each key hashes to a different array index, and
- collection of keys hash to the array index set.

Problem: Key space is often large …

Hash data structure consists of

- 
- 
- 



## Hash Functions

Consists of two parts:

- A hash: function mapping a key to an integer i.
- A compression: function mapping i into the array cells 0 to N-1.

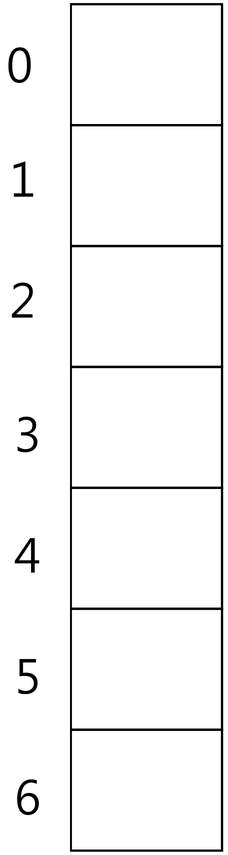
Should has characteristics:

- Computed in ____________.
- Deterministic.
- Satisfy the SUHA.

# Collision handling – Separate chaining

(an example of open hashing)

S = {16, 8, 4, 13, 29, 11, 22}        |S| = n,  h(k) = k%7



# Collision handling – Probe based hashing

(an example of closed hashing)

$S = \{16, 8, 4, 13, 29, 11, 22\}$  $|S| = n$,  $h(k) = k\%7$

| | |
|---|---|
| 0 | |
| 1 | |
| 2 | |
| 3 | |
| 4 | |
| 5 | |
| 6 | |

Try $h(k) = (k + 0)\ \%\ 7$. If full…
try $h(k) = (k + 1)\ \%\ 7$. If full…
try $h(k) = (k + 2)\ \%\ 7$. If full…
try…

# Summary

- Binary search tree
  - FIND, INSERT, DELETE – O_____
- Balanced binary search tree
  - FIND, INSERT, DELETE – O_____
- Hashing
  - FIND, INSERT, DELETE – O_____

  - Disadvantage

    (1) Finding good hash functions is ____________________
    in terms of ____________ and
    ____________

    (2) Poor performance in ______________________

    (3) Not suitable for _____________ due to dynamic resizing of table