2015-1 Programming Language

14. Hash Map

2015년 6월 1일

교수 김 영 탁

영남대학교 공과대학 정보통신공학과

(Tel : +82-53-810-2497; Fax : +82-53-810-4742

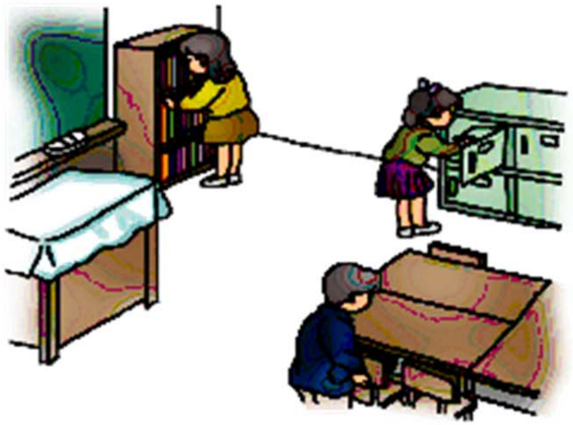
http://antl.yu.ac.kr/; E-mail : ytkim@yu.ac.kr)

Outline

- Hashing 이란?
- 사전 (dictionary), map, table과 해싱
- Hash map의 구현 방법
- 도서관의 서적(book)들을 위한 hash map의 구현

해싱이란?

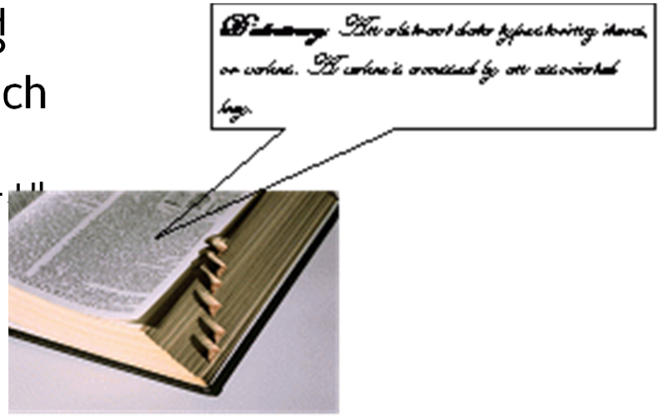
- 대부분의 탐색 (search) 방법들은 키 값 비교 (comparision)로써 탐색하고자 하는 항목에 접근
- 해싱(hashing)
  - 키 값에 대한 산술적 연산에 의해 테이블의 주소를 계산하여 항목에 접근
- 해시 테이블(hash table)
  - 키 값의 연산에 의해 직접 접근이 가능한 구조
- 해싱은 물건을 정리하는 것과 같다



추상자료형 사전구조

◆ 사전구조(dictionary)

- **맵**(map) 또는 **테이블**(table)로 불리움
- 탐색 키와 관련된 값의 2가지 필드로 구성
  - 영어 단어나 사람의 이름 같은 **탐색 키**(search key)
  - 단어의 정의나 주소 또는 전화 번호 같은 탐색 키와 관련된 **값**(value)

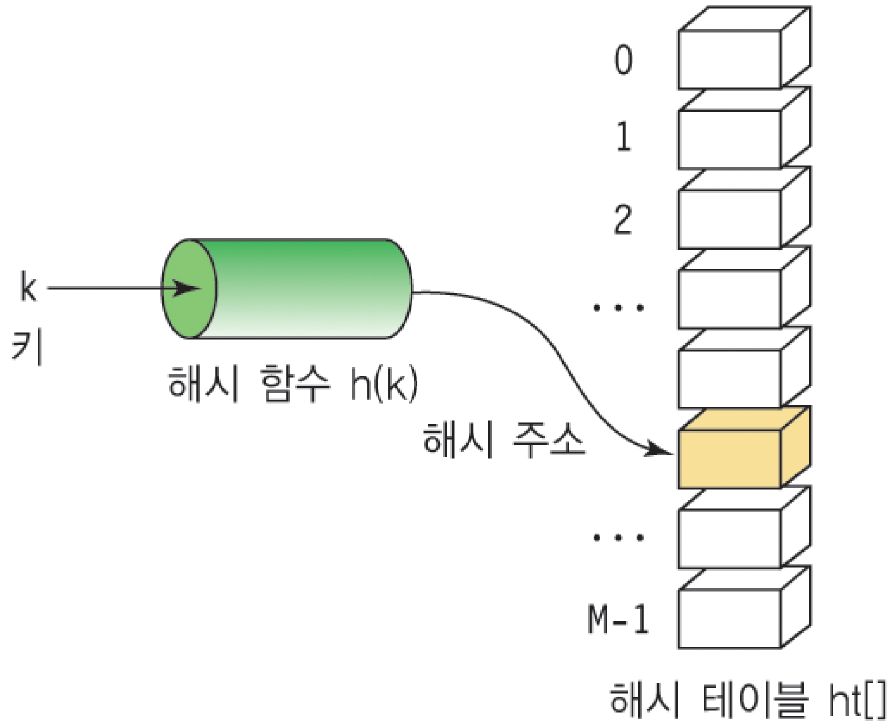


·객체: 일련의 (key,value) 쌍의 집합
·연산:
- add(key, value) ::= (key,value)를 사전에 추가한다.
- delete(key) ::= key에 해당되는 (key,value)를 찾아서 삭제한다.
  관련된 value를 반환한다.만약 탐색이 실패하면 NULL를 반환한다.
- search(key) ::= key에 해당되는 value를 찾아서 반환한다.
  만약 탐색이 실패하면 NULL를 반환한다.

해싱의 구조

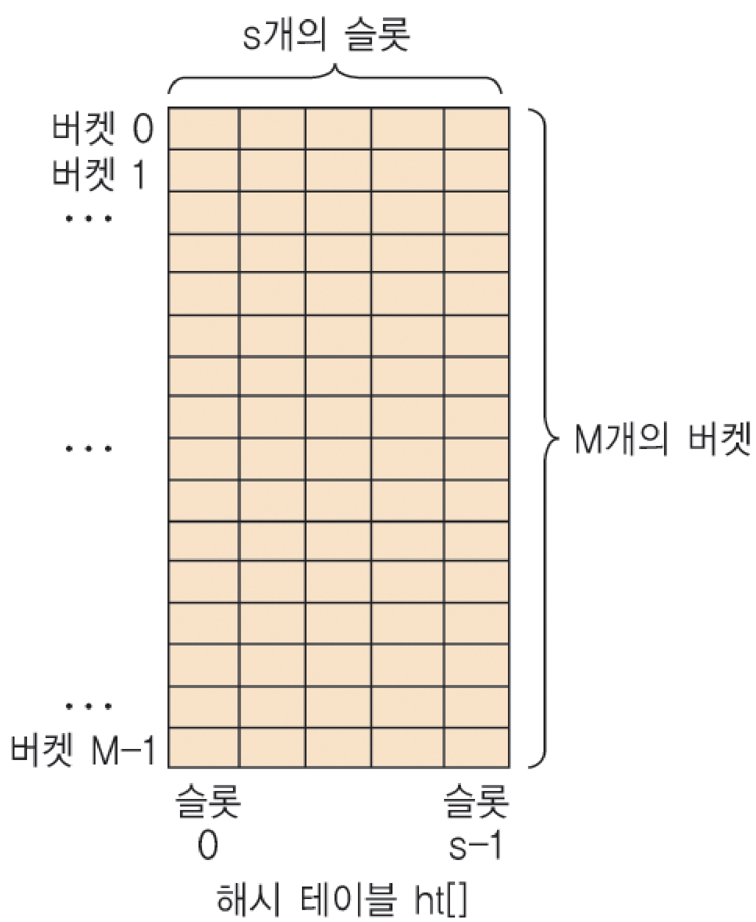
◆ 해시 함수(hash function)

- 탐색키를 입력받아 **해시 코드**(hash code) 및 해시 주소 생성
- 이 해시 주소가 배열로 구현된 **해시 테이블**(hash table)의 인덱스



해시 테이블의 구조

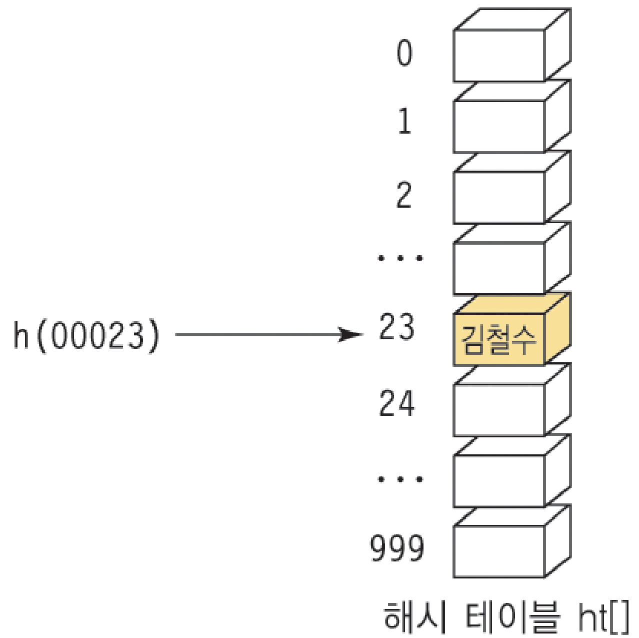
- ◆ **해시테이블 ht**
  - M개의 **버켓**(bucket)으로 구성된 테이블
  - ht[0], ht[1], …,ht[M-1]의 원소를 가짐
  - 각 버켓에 s개의 **슬롯**(slot) 가능
- ◆ **충돌(collision)**
  - 서로 다른 두 개의 탐색키 k1과 k2에 대하여 h(k1) = h(k2)인 경우
- ◆ **오버플로우(overflow)**
  - 충돌이 버켓에 할당된 슬롯 수 보다 많이 발생하는 것
  - 오버플로우 해결 방법 반드시 필요



이상적인 해싱

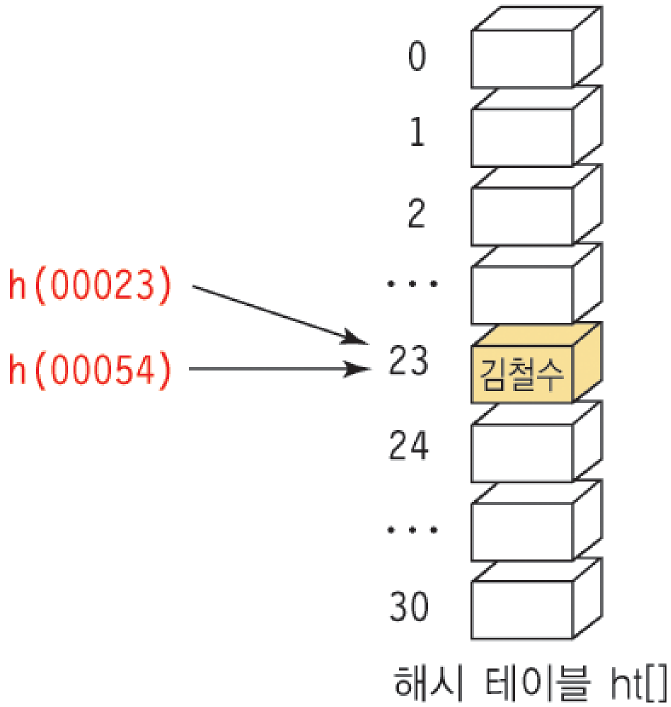
◆ 학생 정보를 해싱으로 저장, 탐색해보자

- 5자리 학번 중에 앞 2자리가 학과 번호, 뒤 3자리가 각 학과의 학생 번호: h(k) = k % 1000;
- 같은 학과 학생들만 가정하면 뒤의 3자리만 사용해서 탐색 가능
- 학번이 00023이라면 이 학생의 인적 사항은 해시테이블 ht[23]에 저장
- 만약 해시테이블이 1000개의 공간을 가지고 있다면 탐색 시간이 O(1)이 되므로 이상적임



실제의 해싱

- 실제로는 해시테이블의 크기가 제한되므로, 존재 가능한 모든 키에 대해
  저장 공간을 할당할 수 없음
- h(k)= k mod M 의 예에서 보듯이 필연적으로 충돌과 오버플로우 발생함
  - M = 31
  - 23 % 31 = 23
  - 54 % 31 = 23



실제의 해싱(cont.)

- **알파벳 문자열 키의 해시함수가 키의 첫 번째 문자의 순서라고 하자**
  - h(keyStr) = keyStr[0] – ‘a’;
- **입력데이터: array, binary, bubble, file, digit, direct, zero, bucket**

  h("array")=0;

  h("binary")=1;

  h("bubble")=1;

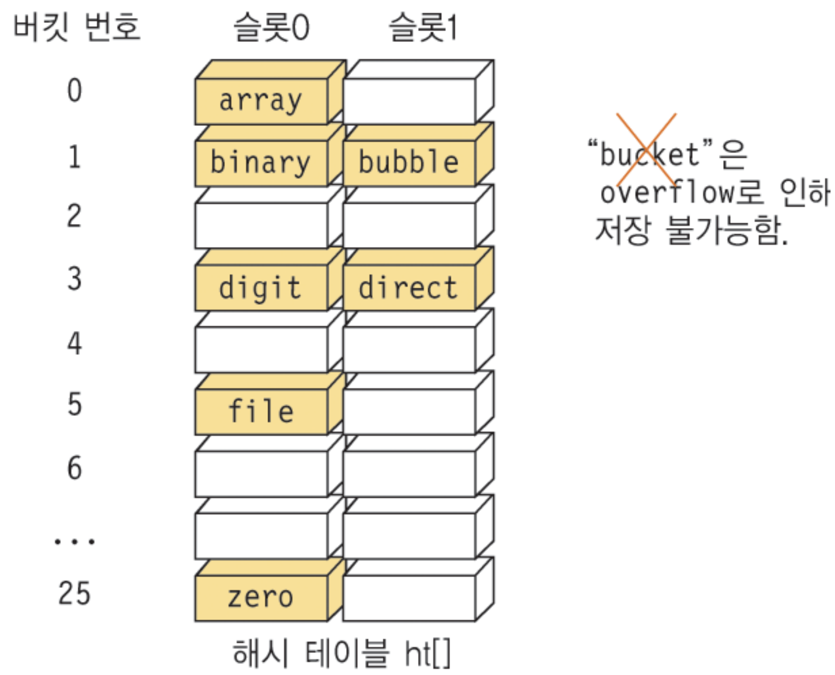
  h(“file”) = 5;

  h(“digit”) = 3;

  h(“direct”) = 3;

  h(“zero”) = 25;

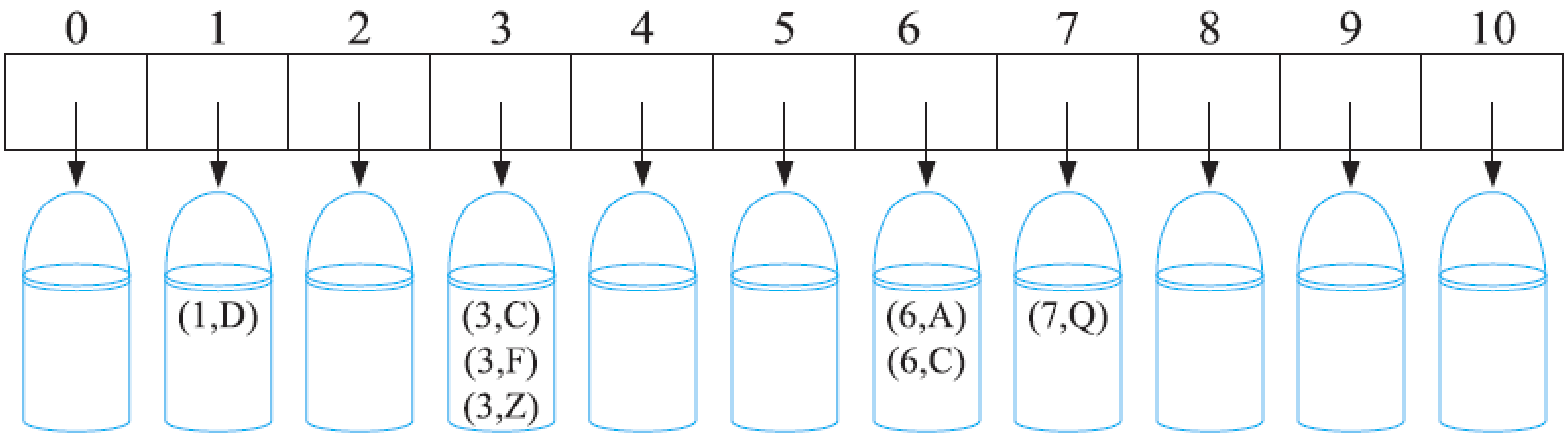
  h("bucket")=1;



Bucket Array 기반의 Hash Map 구현

◆ Bucket Array

- a *bucket array* for a hash table is an array A of size N, where each cell of A is thought of as a bucket (collection of **key-value** pairs), and the integer N defines the capacity of the array



Hash 함수 (1)

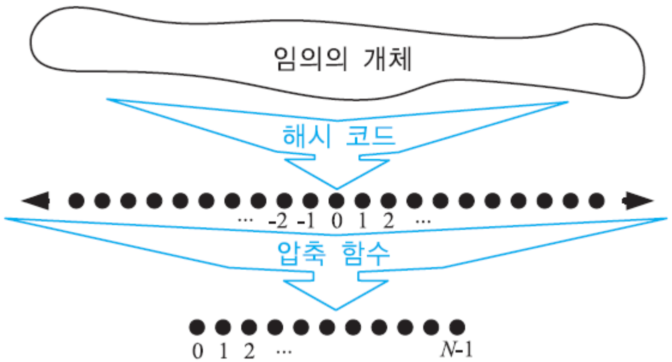
- A hash function is usually specified as the composition of two functions:

  (i) Hash code:

  h_1: keys $\rightarrow$ integers

  (ii) Compression function:

  h_2: integers $\rightarrow [0, N-1]$



- The **hash code** is applied first, and the **compression function** is applied next on the result, i.e.,

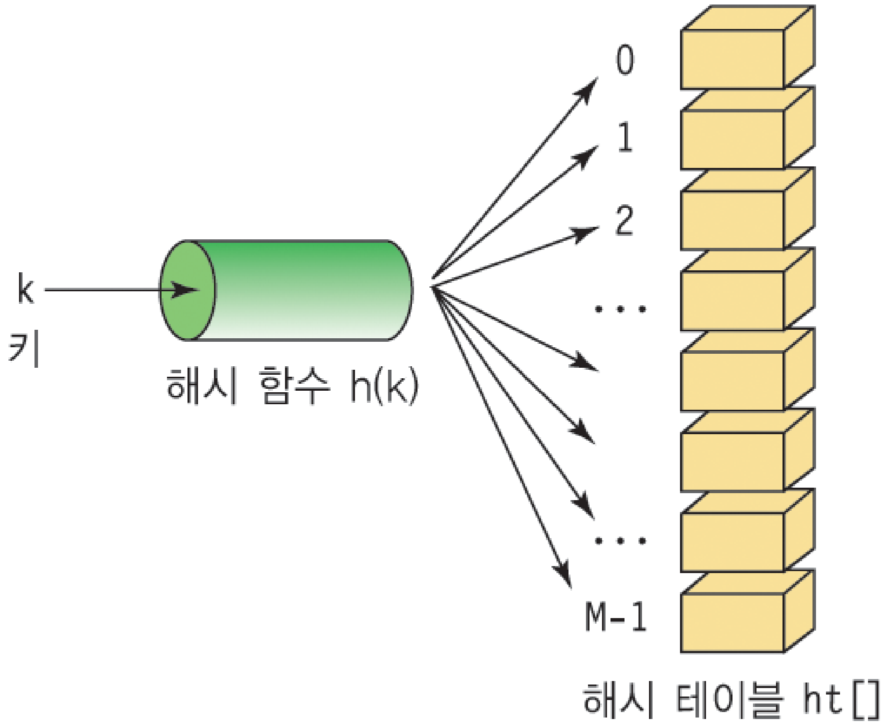
  $$h(x) = h_2(h_1(x))$$

- The goal of the hash function is to "**disperse**" the keys in an apparently random way

Hash 함수 (2)

◆ 좋은 해시 함수의 조건

- 충돌이 적어야 한다
- 해시함수 값이 해시테이블의 주소 영역 내에서 고르게 분포되어야 한다
- 계산이 빨라야 한다



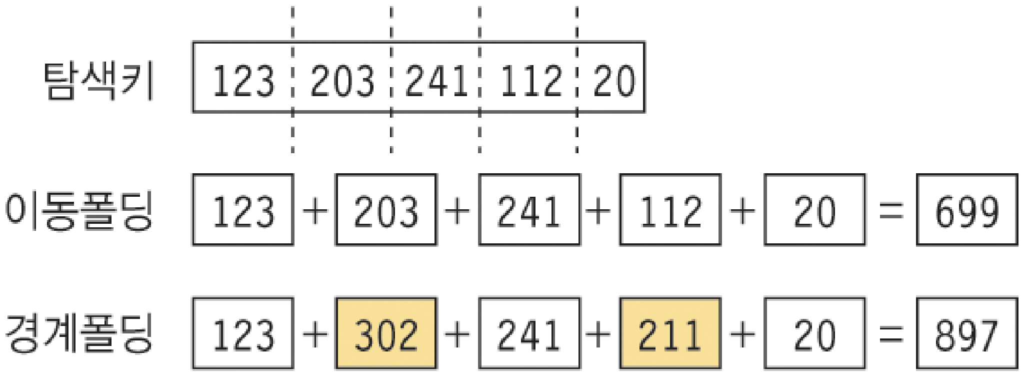
Hash 함수 (3)

◆ 제산 (modular) 함수

- h(k)=k mod M
- 해시 테이블의 크기 M은 **소수**(prime number) (예: 11, 101) 선택

◆ 폴딩 함수

- **이동 폴딩**(shift folding)과 **경계 폴딩**(boundary folding)



Hash 함수 (4)

◆ 중간제곱 함수

- 탐색키를 제곱한 다음, 중간의 몇 비트를 취해서 해시 주소 생성

◆ 비트추출 함수

- 탐색키를 이진수로 간주하여 임의의 위치의 k개의 비트를 해시 주소로 사용

◆ 숫자 분석 방법

- 키 중에서 편중되지 않는 수들을 해시테이블의 크기에 적합하게 조합하여 사용

Hash 함수 (5)

◆ Cyclic Shift Hash Code

```
int hashCode (const char* p, int len)
{
      unsigned int h = 0;
      for (int i=0; i< len; i++)
      {
            h = (h << 5) | (h >> 27);
            h += (unsigned int) p[i];
      }
      return h
}
```

◆ Experimental results

- comparisons of the number of collisions for various shift amounts for 25,000 English words

shift	collisions		shift	collisions	
	total	max		total	max
0	23739	86	9	18	2
1	10517	21	10	277	3
2	2254	6	11	453	4
3	448	3	12	43	2
4	89	2	13	13	2
5	4	2	14	135	3
6	6	2	15	1082	6
7	14	2	16	8760	9
8	105	2			

압축 함수 (Compress Function)

◆ Division

- $h(k) = |k| \bmod N$
- The size N of the hash table is usually chosen to be a prime
- The reason has to do with number theory and is beyond the scope of this course

◆ Multiply, Add and Divide (MAD)

- $h(k) = |ak + b| \bmod N$
- a and b are nonnegative integers such that
  $a \bmod N \neq 0$
- Otherwise, every integer would map to the same value b

충돌해결책

◆ 충돌(collision)

- 서로 다른 탐색 키를 갖는 항목들이 같은 해시 주소를 가지는 현상
- 충돌이 발생하면 해시 테이블에 항목 저장 불가능
- 충돌을 효과적으로 해결하는 방법 반드시 필요

해시테이블

◆ 충돌해결책

- 선형조사법: 충돌이 일어난 항목을 해시 테이블의 다른 위치에 저장
- 체이닝: 각 버켓에 삽입과 삭제가 용이한 연결 리스트 할당

선형조사법(linear probing)

- **충돌이 ht[k]에서 발생했다면,**
  - ht[k+1]이 비어 있는지 조사
  - 만약 비어있지 않다면 ht[k+2] 조사
  - 비어있는 공간이 나올 때까지 계속 조사
  - 테이블의 끝에 도달하게 되면 다시 테이블의 처음부터 조사
  - 조사를 시작했던 곳으로 다시 되돌아오게 되면 테이블이 가득 찬것임
  - 조사되는 위치: h(k), h(k)+1, h(k)+2,…

- **군집화(clustering)과 결합(Coalescing) 문제 발생**

선형조사법(linear probing)

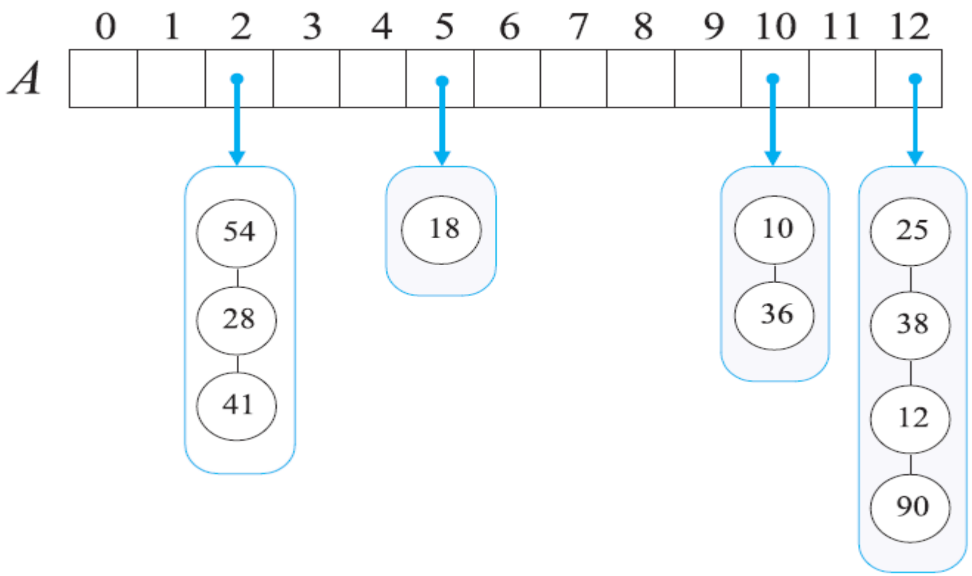
- (예) h(k)=k mod 7

1단계 (8) : h(8) = 8 mod 7 = 1(저장)
2단계 (1) : h(1) = 1 mod 7 = 1(충돌발생)
(h(1)+1) mod 7 = 2(저장)
3단계 (9) : h(9) = 9 mod 7 = 2(충돌발생)
(h(9)+1) mod 7 = 3(저장)
4단계 (6) : h(6) = 6 mod 7 = 6(저장)
5단계 (13) : h(13) = 13 mod 7 = 6(충돌 발생)
(h(13)+1) mod 7 = 0(저장)

	1단계	2단계	3단계	4단계	5단계
[0]					13
[1]	8	8	8	8	8
[2]		1	1	1	1
[3]			9	9	9
[4]					
[5]					
[6]				6	6

Chaining을 사용한 충돌 해결

- Collisions occur when different elements are mapped to the same cell
- Separate Chaining: let each cell in the table point to a linked list of entries that map there
- Separate chaining is simple, but requires additional memory outside the table



체이닝 (chaining)

◆ 오버플로우 문제를 연결 리스트 (linked list) 로 해결

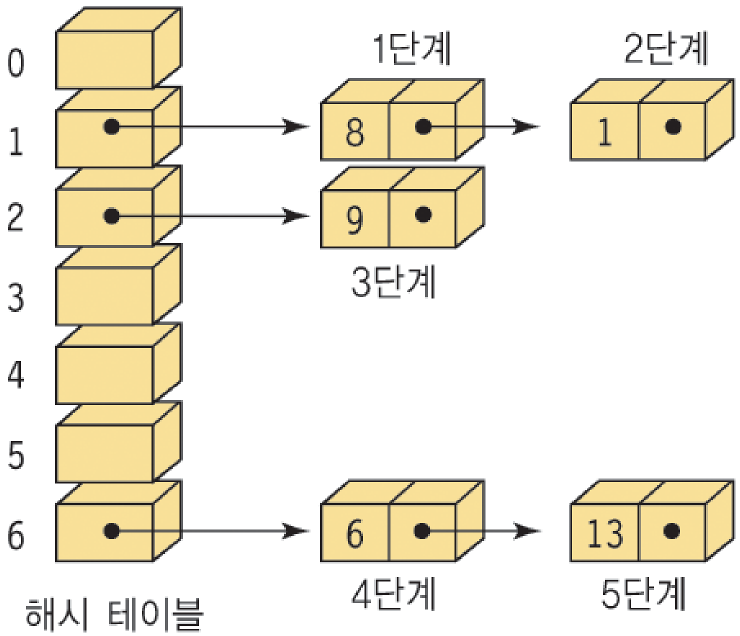
- 각 버켓에 고정된 슬롯이 할당되어 있지 않음
- 각 버켓에, 삽입과 삭제가 용이한 연결 리스트 할당
- 버켓 내에서는 연결 리스트 순차 탐색

◆ (예) 크기가 7인 해시테이블에서

- h(k)=k mod 7의 해시 함수 사용
- 입력 (8, 1, 9, 6, 13) 적용

체이닝(chaining)

1단계 (8) : h(8) = 8 mod 7 = 1(저장)
2단계 (1) : h(1) = 1 mod 7 = 1(충돌발생->새로운 노드 생성 저장)
3단계 (9) : h(9) = 9 mod 7 = 2(저장)
4단계 (6) : h(6) = 6 mod 7 = 6(저장)
5단계 (13) : h(13) = 13 mod 7 = 6(충돌 발생->새로운 노드 생성 저장)



해싱과 다른 탐색 방법의 비교

탐색 방법		탐색	삽입	삭제
순차 탐색		$O(n)$	$O(1)$	$O(n)$
이진 탐색		$O(\log_2 n)$	$O(\log_2 n + n)$	$O(\log_2 n + n)$
이진 탐색 트리	균형 트리	$O(\log_2 n)$	$O(\log_2 n)$	$O(\log_2 n)$
	경사 트리	$O(n)$	$O(n)$	$O(n)$
해싱	최선의 경우	$O(1)$	$O(1)$	$O(1)$
	최악의 경우	$O(n)$	$O(n)$	$O(n)$

Re-hashing

◆ Resizing

- if the load factor of a hash table goes significantly above a specified threshold, then it is common to resize the hash table with the new array's size be at least double the previous size
- once we have allocated a new bucket array, we must define a new hash function to go with it
- given this new hash function, we then reinsert every item from the old array into the new array using this new hash function

도서관의 서적 (Book) 들을 위한 Hash Map 구현

```
/* Book.h */

#ifndef BOOK_H
#define BOOK_H

typedef struct Date {
    int year;
    int month;
    int day;
} Date;

typedef struct Book {
    int isbn;
    char title[16];
    char author_name[16];
    double price;
    char publisher[16];
    Date publication_date;
} Book;

Book *genBook(int isbn);
void printBook(Book *pBk);
#endif
```

```
/* Book.cpp (1) */
#include <iostream>
#include <stdio.h>
#include "Book.h"

using namespace std;
extern void genBigRandArray(int mA[], int range);

Book *genBook(int isbn)
{
    int name_len;
    int j;
    double br;
    Book *pBk;

    pBk = (Book *)malloc(sizeof(Book));
    pBk->isbn = isbn;

    name_len = rand() % 11 + 5;
    pBk->title[0] = rand() % 26 + 'A';
    for (j = 1; j<name_len; j++)
        pBk->title[j] = rand() % 26 + 'a';
    pBk->title[j] = '₩0';
```

```
/* Book.cpp (2) */

    br = ((rand() << 15 | rand()) % 100001) / 100.0;
    pBk->price = br;

    name_len = rand() % 11 + 5;
    pBk->publisher[0] = rand() % 26 + 'A';
    for (j = 1; j<name_len; j++)
        pBk->publisher[j] = rand() % 26 + 'a';
    pBk->publisher[j] = '₩0';

    pBk->publication_date.year = rand() % 41 + 1960;
    pBk->publication_date.month = rand() % 12 + 1;
    pBk->publication_date.day = rand() % 30 + 1;

    return pBk;
}

void printBook(Book *pBk)
{
    printf("book[ isbn: %7d, title: %-15s, author: %-15s", pBk->isbn, pBk->title,
      pBk->author_name);
    printf(", price: %7.2f, , Publisher: %-15s", pBk->price, pBk->publisher);
    printf(", Publication date: %4d-%2d-%2d ]", pBk->publication_date.year,
     pBk->publication_date.month, pBk->publication_date.day);
}
```

```
/* Entry.h */

#ifndef ENTRY_H
#define ENTRY_H
#include <string>
#include "Book.h"
using namespace std;
#define Key char*
typedef struct Entry
{
    Key pKey; // key string
    Book *pBk;
} Entry;

void printEntry(Entry *pE);
#endif
```

```
/* Entry.cpp */

#include "Entry.h"
#include "Book.h"

void printEntry(Entry *pE)
{
    printf("[ key(title): %-15s, ", pE->pKey);
    printBook(pE->pBk);
    printf(" ], ");
}
```

```
/* Bucket.h */
#ifndef BUCKET_H
#define BUCKET_H

#include "Entry.h"

typedef struct ListNode
{
    Entry *pE;
    ListNode* pPrev;
    ListNode* pNext;
} ListNode;

typedef struct Bucket {
    ListNode *pFirst;
    ListNode *pLast;
    int num_node;
} Bucket;

void initBucket(Bucket *pBkt);
int insertBucket(Bucket *pBkt, Entry *pE);
void printBucket(Bucket *pBkt);
bool searchBucket(Bucket *pBkt, Entry *pE);
void deleteBucket(Bucket *pBkt);
#endif
```

```
/* Bucket.cpp (1) */
#include "Bucket.h"

void initBucket(Bucket *pBkt)
{
    pBkt->num_node = 0;
    pBkt->pFirst = pBkt->pLast = NULL;
}

int insertBucket(Bucket *pBkt, Entry *pE)
{
    ListNode *pNew;

    pNew = (ListNode *)malloc(sizeof(ListNode));
    pNew->pNext = pNew->pPrev = NULL;
    pNew->pE = pE;
    if (pBkt->num_node == 0)
    {
        pBkt->pFirst = pBkt->pLast = pNew;
        pBkt->num_node++;
    }
    else {
        pNew->pPrev = pBkt->pLast;
        pBkt->pLast->pNext = pNew;
        pBkt->pLast = pNew;
        pNew->pNext = NULL;
        pBkt->num_node++;
    }
    return pBkt->num_node;
}
```

```
/* Bucket.cpp (2) */

void printBucket(Bucket *pBkt)
{
    Entry *pE;
    int num_entry = pBkt->num_node;
    ListNode *pLN = pBkt->pFirst;
    if (pLN != NULL)
    {
        printf("₩n");
        for (int i = 0; (i < num_entry)
             && (pLN != NULL); i++)
        {
            pE = pLN->pE;
            if (pE != NULL) {
                printf("   ");
                printEntry(pE);
                printf("₩n");
            }
            else {
                printf("Error in printEntry()
                       pE is NULL !!₩n");
                exit;
            }
            pLN = pLN->pNext;
        }
    }
}
```

```
/* Bucket.cpp (3) */

bool searchBucket(Bucket *pBkt, Entry *pE)
{
    ListNode *pLN;
    Entry *pCurE;
    int num_entry = pBkt->num_node;
    pLN = pBkt->pFirst;
    for (int i = 0; (i < num_entry) && (pLN != NULL); i++)
    {
        pCurE = pLN->pE;
        if ((pE != NULL) && (pCurE->pBk->isbn
                 == pE->pBk->isbn))
        {
            return true;
        }
        pLN = pLN->pNext;
    }
    return false;

}
```

```
/* Bucket.cpp (4) */

void deleteBucket(Bucket *pBkt)
{
    ListNode *pLN, *pCurLN;
    int num_entry = pBkt->num_node;
    pLN = pBkt->pFirst;
    for (int i = 0; (i < num_entry) && (pLN != NULL); i++)
    {
        pCurLN = pLN;
        pLN = pLN->pNext;
        free(pCurLN->pE);
        free(pCurLN);
    }
}
```

```
/* Hash.h */

#ifndef HASH_H
#define HASH_H
#include "Entry.h" // Key is defined in Entry.h

unsigned int cyclicShiftHashCode(Key keyStr);

#endif
```

```
/* Hash.cpp */
#include "Hash.h"
#include "Entry.h"

unsigned int cyclicShiftHashCode(Key keyStr)
{
    unsigned int h = 0;
    int len = strlen(keyStr);
    for (int i = 0; i < len; i++)
    {
        h = (h << 5) | (h >> 27);
        h += (unsigned int)keyStr[i];
    }
    return h;
}
```

```c
/* HashMap.h */

#ifndef HASHMAP_H
#define HASHMAP_H

#include <stdio.h>
#include "Bucket.h"
#include "Entry.h"
#include "Hash.h"

typedef struct HashMap
{
    int bktArraySize; // Size of Bucket Array
    Bucket *bktArray; // BucketArray[]
    unsigned int(*hash)(Key);
    HashMap(int bktArrSize); // constructor of HashMap
} HashMap;

void printHashMap(HashMap *pHM);
void insertHashMap(HashMap *pHM, Entry *pE);
void searchHashMap(HashMap *pHM, Entry *pE);
void deleteHashMap(HashMap *pHM);
#endif
```

```
/* HashMap.cpp (1) */

#include "HashMap.h"
#include <stdlib.h>

HashMap::HashMap(int bktArrSize)
{
    bktArraySize = bktArrSize;
    bktArray = (Bucket *)malloc(sizeof(Bucket)* bktArraySize);
    for (int i = 0; i < bktArraySize; i++)
    {
        initBucket(&bktArray[i]);
    }

}

void insertHashMap(HashMap *pHM, Entry *pE)
{
    unsigned int hashCode;
    unsigned int bucketIndex;
    Bucket *pBkt;

    hashCode = pHM->hash(pE->pKey);
    bucketIndex = hashCode % pHM->bktArraySize;
    pBkt = &pHM->bktArray[bucketIndex];
    insertBucket(pBkt, pE);
}
```

```
/* HashMap.cpp (2) */

void printHashMap(HashMap *pHM)
{
    Bucket *pBkt;
    Entry *pE;
    int bktIndex;

    for (bktIndex = 0; bktIndex < pHM->bktArraySize; bktIndex++)
    {
        pBkt = &pHM->bktArray[bktIndex];
        printf("Bucket[%2d]: %2d entries: ", bktIndex, pBkt->num_node);
        printBucket(pBkt);
        printf("₩n");
    }
}
```

```
/* HashMap.cpp (3) */

void searchHashMap(HashMap *pHM, Entry *pE)
{
    unsigned int hashCode;
    unsigned int bucketIndex;
    Bucket *pBkt;

    hashCode = pHM->hash(pE->pKey);
    bucketIndex = hashCode % pHM->bktArraySize;
    pBkt = &pHM->bktArray[bucketIndex];
    if (searchBucket(pBkt, pE))
    {
        printf("==> found in %2d_th bucket₩n", bucketIndex);
    } else {
        printf("Failed to find the entry from %2dd_th bucket₩n", bucketIndex);
    }
}

void deleteHashMap(HashMap *pHM)
{
    Bucket *pBkt;
    int bktIndex;
    for (bktIndex = 0; bktIndex < pHM->bktArraySize; bktIndex++)
    {
        pBkt = &pHM->bktArray[bktIndex];
        deleteBucket(pBkt);
    }
}
```

```
/* main.cpp (1) */

#include <stdio.h>
#include <stdlib.h>
#include "HashMap.h"
#include "Hash.h"
#include "Star.h"
#include "Book.h"

#define NUM_BOOK 30
#define BUCKET_ARRAY_SIZE 17
extern void genBigRandArray(int mA[], int range);
void main()
{
    int *starID;
    int *bookISBN;
    Book *books[NUM_BOOK], *pB;
    Entry *pE;
    HashMap hashMap((int)BUCKET_ARRAY_SIZE);
    hashMap.hash = cyclicShiftHashCode;

    bookISBN = (int *)malloc(sizeof(int)* NUM_BOOK);
    genBigRandArray(bookISBN, NUM_BOOK);
    printf("Generate and insert %d books into hash map ...₩n", NUM_BOOK);
```

```
/* main.cpp (2) */

     for (int i = 0; i < NUM_BOOK; i++)
    {
        books[i] = genBook(bookISBN[i]);
        pE = (Entry *)malloc(sizeof(Entry));
        pE->pBk= books[i];
        pE->pKey = books[i]->title;
        insertHashMap(&hashMap, pE);
    }
    printf("Hash map status: ₩n");
    printHashMap(&hashMap);

    printf("Searching star from the Hash Map ...₩n");
    for (int i = 0; i < NUM_BOOK; i++)
    {
        pB = books[i];
        pE = (Entry *)malloc(sizeof(Entry));
        pE->pBk = books[i];
        pE->pKey = books[i]->title;
        printBook(pB);
        printf(" : ");
        searchHashMap(&hashMap, pE);
        free(pE);
    }
    deleteHashMap(&hashMap);
    free(bookISBN);
}
```

◆ 실행결과 (1)

```
No duplication in numers generated by BigRand() for 30 random numbers !!
Generate and insert 30 books into hash map ...
Hash map status:
Bucket[ 0]:  2 entries:
   [ key(title): Fzotcjtnzxuglsd, book[ isbn:     22, title: Fzotcjtnzxuglsd, author: Mzcnockvfajfr  , price:  747.02, , Publisher: Thowkb        , Publication date: 1963- 8-15 ] ],
   [ key(title): Mjrmbsztsshf   , book[ isbn:      9, title: Mjrmbsztsshf   , author: Oefwsjrxjh     , price:  804.01, , Publisher: Yupzw         , Publication date: 1996- 1-23 ] ],
Bucket[ 1]:  1 entries:
   [ key(title): Ydhacwyhsgewz  , book[ isbn:      6, title: Ydhacwyhsgewz  , author: Tgonzltjhgauh  , price:  955.38, , Publisher: Reggj         , Publication date: 2000- 9-19 ] ],
Bucket[ 2]:  3 entries:
   [ key(title): Qaybnefxnxv    , book[ isbn:      2, title: Qaybnefxnxv    , author: Zedyyhngy      , price:  900.23, , Publisher: Udmphmec      , Publication date: 1970-11-18 ] ],
   [ key(title): Mbmxrmhuyf     , book[ isbn:     23, title: Mbmxrmhuyf     , author: Qgajqkckl      , price:  237.61, , Publisher: Yxqkq         , Publication date: 1964- 7-10 ] ],
   [ key(title): Nhhssqctydteam , book[ isbn:     17, title: Nhhssqctydteam , author: Cjbprhtnegy    , price:  528.47, , Publisher: Gcjwlgrsme    , Publication date: 1963- 1-15 ] ],
Bucket[ 3]:  5 entries:
   [ key(title): Srtek          , book[ isbn:     15, title: Srtek          , author: Qdcyzjeeuhmsrq , price:  195.47, , Publisher: Ijipfionee    , Publication date: 1964- 8-28 ] ],
   [ key(title): Duburis        , book[ isbn:      4, title: Duburis        , author: Tbrecuyk       , price:  866.75, , Publisher: Cvkdz         , Publication date: 1979-10-20 ] ],
   [ key(title): Xdjgkjelrlpax  , book[ isbn:      8, title: Xdjgkjelrlpax  , author: Mceroswitdp    , price:  880.64, , Publisher: Clifkeljyt    , Publication date: 1965- 8- 4 ] ],
   [ key(title): Pzsqmet        , book[ isbn:     20, title: Pzsqmet        , author: Gepspxnvji     , price:  193.38, , Publisher: Lyynmkmnuvk   , Publication date: 1963-12- 5 ] ],
   [ key(title): Smtpjhaefqz    , book[ isbn:     28, title: Smtpjhaefqz    , author: Auldrchjccdyrf , price:  857.70, , Publisher: Ivuyeegfivdrc , Publication date: 1971- 1- 9 ] ],
Bucket[ 4]:  1 entries:
   [ key(title): Cdwrac         , book[ isbn:      3, title: Cdwrac         , author: Fmzkg          , price:  493.30, , Publisher: Fodkjmjqwi    , Publication date: 1996- 8-23 ] ],
Bucket[ 5]:  2 entries:
   [ key(title): Myubzazcp      , book[ isbn:     13, title: Myubzazcp      , author: Hktkydz        , price:  613.58, , Publisher: Uypurf        , Publication date: 1973- 6-15 ] ],
   [ key(title): Gekyrgzvxdhpo  , book[ isbn:     21, title: Gekyrgzvxdhpo  , author: Mvafyr         , price:  129.79, , Publisher: Svkhtq        , Publication date: 1976- 5- 8 ] ],
Bucket[ 6]:  4 entries:
   [ key(title): Dmbppweq       , book[ isbn:     26, title: Dmbppweq       , author: Gjoparm        , price:  946.57, , Publisher: Dqyoxyt       , Publication date: 1982- 4-20 ] ],
   [ key(title): Awdydcp        , book[ isbn:     10, title: Awdydcp        , author: Jbxphoohpkw    , price:  206.23, , Publisher: Hrqzhnbnfuvqnqq, Publication date: 1993- 2-16 ] ],
   [ key(title): Wtvjs          , book[ isbn:      7, title: Wtvjs          , author: Baoio          , price:  208.39, , Publisher: Hypnvruihoswki , Publication date: 1970- 9-17 ] ],
   [ key(title): Gdnmphnqkamhurk, book[ isbn:     24, title: Gdnmphnqkamhurk, author: Rffaclvgr      , price:  640.19, , Publisher: Ldacll        , Publication date: 1988- 7-29 ] ],
Bucket[ 7]:  1 entries:
   [ key(title): Qdredakubn     , book[ isbn:      1, title: Qdredakubn     , author: Guproqylobcw   , price:   88.29, , Publisher: Zmausjgm      , Publication date: 1992- 3- 1 ] ],
Bucket[ 8]:  2 entries:
   [ key(title): Ixwtbvtrehbbc  , book[ isbn:     14, title: Ixwtbvtrehbbc  , author: Xifbxvfbcg     , price:  548.05, , Publisher: Qckcotz       , Publication date: 1988- 1-29 ] ],
   [ key(title): Iduku          , book[ isbn:     25, title: Iduku          , author: Jzefczzzb      , price:  271.07, , Publisher: Dpqzikfobuc   , Publication date: 1999-12-14 ] ],
Bucket[ 9]:  0 entries: Bucket[10]:  2 entries:
   [ key(title): Cwljfrimpmy    , book[ isbn:     18, title: Cwljfrimpmy    , author: Chzriw         , price:  363.53, , Publisher: Rxbgfcbcey    , Publication date: 1967- 4-13 ] ],
   [ key(title): Rsigbhz        , book[ isbn:     11, title: Rsigbhz        , author: Zujxmmyspnara  , price:   81.83, , Publisher: Egjcc         , Publication date: 1960- 8- 2 ] ],
Bucket[11]:  2 entries:
   [ key(title): Zrnavymmtatbdz , book[ isbn:     12, title: Zrnavymmtatbdz , author: Soemuvnpppsuacb, price:  931.53, , Publisher: Xmhecthlegr   , Publication date: 1967- 5- 8 ] ],
   [ key(title): Wospofghfo     , book[ isbn:      0, title: Wospofghfo     , author: Qvlqfxwwkmfxd  , price:  820.21, , Publisher: Mdcaszsgovsodkj, Publication date: 1980-12- 9 ] ],
Bucket[12]:  2 entries:
   [ key(title): Jvbjtsqdjo     , book[ isbn:     29, title: Jvbjtsqdjo     , author: Tgpknfpf       , price:  518.50, , Publisher: Fieowq        , Publication date: 1965-11- 3 ] ],
   [ key(title): Qhimvfvuzwyvi  , book[ isbn:     27, title: Qhimvfvuzwyvi  , author: Gfullkjduhs    , price:  659.64, , Publisher: Btlkmfqrm     , Publication date: 1979- 6-22 ] ],
Bucket[13]:  1 entries:
   [ key(title): Urpixiqflduuveo, book[ isbn:     16, title: Urpixiqflduuveo, author: Wqcudhnefnj    , price:  427.08, , Publisher: Muczf         , Publication date: 1974- 1-29 ] ],
Bucket[14]:  0 entries: Bucket[15]:  1 entries:
   [ key(title): Pxiogvliexdzuz , book[ isbn:      5, title: Pxiogvliexdzuz , author: Srkrusv        , price:  447.15, , Publisher: Rzmwzpowkjilef , Publication date: 1974- 9- 9 ] ],
Bucket[16]:  1 entries:
   [ key(title): Digpnpuuhg     , book[ isbn:     19, title: Digpnpuuhg     , author: Pqnjwjmwax     , price:  232.34, , Publisher: Snhhlqqrz     , Publication date: 1996- 2- 8 ] ],
```

◆ 실행결과 (2)

```
Searching star from the Hash Map ...
book[ isbn:      15, title: Srtek          , author: Qdcyzjeeuhmsrq , price:  195.47, , Publisher: Ijipfionee     , Publication date: 1964- 8-28 ] : ==> found in  3_th bucket
book[ isbn:      12, title: Zrnavymmtatbdz , author: Soemuvnpppsuacb, price:  931.53, , Publisher: Xmhecthlegr    , Publication date: 1967- 5- 8 ] : ==> found in 11_th bucket
book[ isbn:      26, title: Dmbppweq       , author: Gjoparm        , price:  946.57, , Publisher: Dqyoxyt        , Publication date: 1982- 4-20 ] : ==> found in  6_th bucket
book[ isbn:      10, title: Awdydcp        , author: Jbxphoohpkw    , price:  206.23, , Publisher: Hrqzhnbnfuvgnqq, Publication date: 1993- 2-16 ] : ==> found in  6_th bucket
book[ isbn:       5, title: Pxiogvliexdzuz , author: Srkrusv        , price:  447.15, , Publisher: Rzmwzpowkjilef , Publication date: 1974- 9- 9 ] : ==> found in 15_th bucket
book[ isbn:      19, title: Digpnpuuhg     , author: Pqnjwjmwax     , price:  232.34, , Publisher: Snhhlqqrz      , Publication date: 1996- 2- 8 ] : ==> found in 16_th bucket
book[ isbn:      22, title: Fzotcjtnzxuglsd, author: Mzcnockvfajfr  , price:  747.02, , Publisher: Thowkb         , Publication date: 1963- 8-15 ] : ==> found in  0_th bucket
book[ isbn:      18, title: Cwljfrimpmy    , author: Chzriw         , price:  363.53, , Publisher: Rxbgfcbcey     , Publication date: 1967- 4-13 ] : ==> found in 10_th bucket
book[ isbn:      14, title: Ixwtbvtrehbbc  , author: Xifbxvfbcg     , price:  548.05, , Publisher: Qckcotz        , Publication date: 1988- 1-29 ] : ==> found in  8_th bucket
book[ isbn:       9, title: Mjrmbsztsshf   , author: Oefwsjrxjh     , price:  804.01, , Publisher: Yupzw          , Publication date: 1996- 1-23 ] : ==> found in  0_th bucket
book[ isbn:      16, title: Urpixiqflduuveo, author: Wqcudhnefnj    , price:  427.08, , Publisher: Muczf          , Publication date: 1974- 1-29 ] : ==> found in 13_th bucket
book[ isbn:       4, title: Duburis        , author: Tbrecuyk       , price:  866.75, , Publisher: Cvkdz          , Publication date: 1979-10-20 ] : ==> found in  3_th bucket
book[ isbn:      25, title: Iduku          , author: Jzefczzzb      , price:  271.07, , Publisher: Dpqzikfobuc    , Publication date: 1999-12-14 ] : ==> found in  8_th bucket
book[ isbn:       8, title: Xdjgkjelrlpax  , author: Mceroswitdp    , price:  880.64, , Publisher: Clifkeljyt     , Publication date: 1965- 8- 4 ] : ==> found in  3_th bucket
book[ isbn:       2, title: Qaybnefxnxv    , author: Zedyyhngy      , price:  900.23, , Publisher: Udmphmec       , Publication date: 1970-11-18 ] : ==> found in  2_th bucket
book[ isbn:       0, title: Wospofghfo     , author: Qvlqfxwwkmfxd  , price:  820.21, , Publisher: Mdcaszsgovsodkj, Publication date: 1980-12- 9 ] : ==> found in 11_th bucket
book[ isbn:      23, title: Mbmxrmhuyf     , author: Qgajqkckl      , price:  237.61, , Publisher: Yxqkq          , Publication date: 1964- 7-10 ] : ==> found in  2_th bucket
book[ isbn:      13, title: Myubzazcp      , author: Hktkydz        , price:  613.58, , Publisher: Uypurf         , Publication date: 1973- 6-15 ] : ==> found in  5_th bucket
book[ isbn:      21, title: Gekyrgzvxdhpo  , author: Mvafyr         , price:  129.79, , Publisher: Svkhtq         , Publication date: 1976- 5- 8 ] : ==> found in  5_th bucket
book[ isbn:      11, title: Rsigbhz        , author: Zujxmmyspnara  , price:   81.83, , Publisher: Egjcc          , Publication date: 1960- 8- 2 ] : ==> found in 10_th bucket
book[ isbn:      29, title: Jvbjtsqdjo     , author: Tgpknfpf       , price:  518.50, , Publisher: Fieowq         , Publication date: 1965-11- 3 ] : ==> found in 12_th bucket
book[ isbn:      20, title: Pzsqmet        , author: Gepspxnvji     , price:  193.38, , Publisher: Lyynmkmnuvk    , Publication date: 1963-12- 5 ] : ==> found in  3_th bucket
book[ isbn:       3, title: Cdwrac         , author: Fmzkg          , price:  493.30, , Publisher: Fodkjmjqwi     , Publication date: 1996- 8-23 ] : ==> found in  4_th bucket
book[ isbn:      27, title: Qhimvfvuzwyvi  , author: Gfullkjduhs    , price:  659.64, , Publisher: Btlkmfqrm      , Publication date: 1979- 6-22 ] : ==> found in 12_th bucket
book[ isbn:      17, title: Nhhssqctydteam , author: Cjbprhtnegy    , price:  528.47, , Publisher: Gcjwlgrsme     , Publication date: 1963- 1-15 ] : ==> found in  2_th bucket
book[ isbn:       7, title: Wtvjs          , author: Baoio          , price:  208.39, , Publisher: Hypnvruihoswki , Publication date: 1970- 9-17 ] : ==> found in  6_th bucket
book[ isbn:       6, title: Ydhacwyhsgewz  , author: Tgonzltjhgauh  , price:  955.38, , Publisher: Reqgj          , Publication date: 2000- 9-19 ] : ==> found in  1_th bucket
book[ isbn:      28, title: Smtpjhaefqz    , author: Auldrchjccdyrf , price:  857.70, , Publisher: Ivuyeegfivdrc  , Publication date: 1971- 1- 9 ] : ==> found in  3_th bucket
book[ isbn:       1, title: Qdredakubn     , author: Guproqylobcw   , price:   88.29, , Publisher: Zmausjgm       , Publication date: 1992- 3- 1 ] : ==> found in  7_th bucket
book[ isbn:      24, title: Gdnmphnqkamhurk, author: Rffaclvgr      , price:  640.19, , Publisher: Ldacll         , Publication date: 1988- 7-29 ] : ==> found in  6_th bucket
계속하려면 아무 키나 누르십시오 . . .
```

Homework 14

14.1 별 (star)들을 관리하기 위한 hash map 구현

(1) Star들을 관리하기 위한 구조체 Star는 다음과 같이 정의된다.

```
typedef struct Star {
    char name[STAR_NAME_LEN];
    int id;
    double distance;
    double luminosity;
    double mass;
    double radius;
    int age;
} Star;
```

(2) 구조체 Star 관련 함수로 genStar()와 printStar() 함수를 작성하라.

- Star* genStar(int starID);
- void printStar(Star *pS)

(3) hash map을 구성하기 위한 구조체 Entry, ListNode, Bucket, HashMap을 작성하고, 관련 함수들을 작성하라. hash 함수로는 cyclic shift hash code를 사용할 것.

(4) 총 90개의 star를 생성한 후, bucket array size 101인 hash map에 star들을 추가한 후, 각 bucket에서의 충돌 발생 회수를 비교하라.

(5) hash map에 포함된 각 star들을 searchHashMap(HashMap *pHM, Entry *pE) 함수를 사용하여 탐색하고, 각 star가 정확하게 탐색되는가를 확인하라.