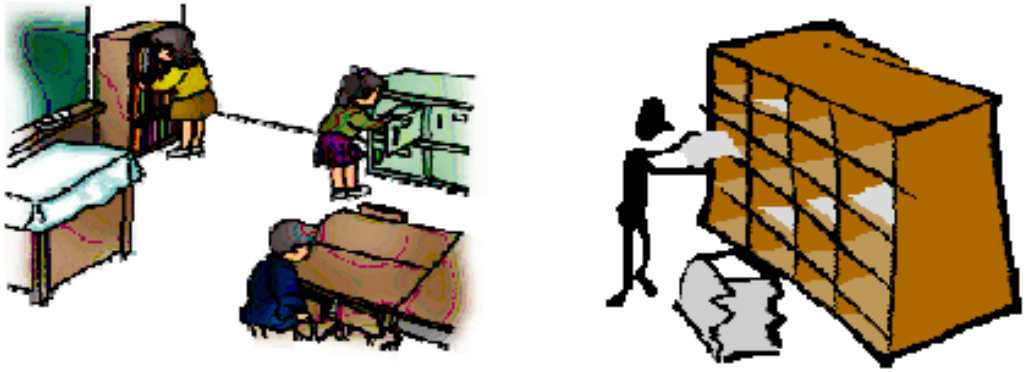
MYONGJI UNIVERSITY

Hash

Dr. Seung Chul Han
Dept. Computer Engineering
Myongji University

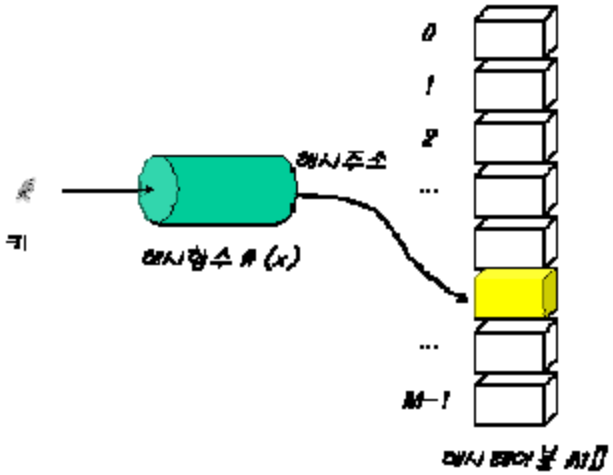
Hash

- 지금까지 배운 모든 탐색 방법들은 키값을 비교함으로써 탐색하고자 하는 항목에 접근
- 해싱은 키 값에 직접 산술적인 연산을 적용하여 항목이 저장되어 있는 테이블의 주소를 계산하여 항목에 접근
- 값의 연산에 의해 직접 접근이 가능한 구조를 **해시 테이블**(hash table)이라 부르며, 해시테이블을 이용한 탐색을 **해싱**(hashing)
- 해싱의 예: 심볼 테이블



해싱의 구조

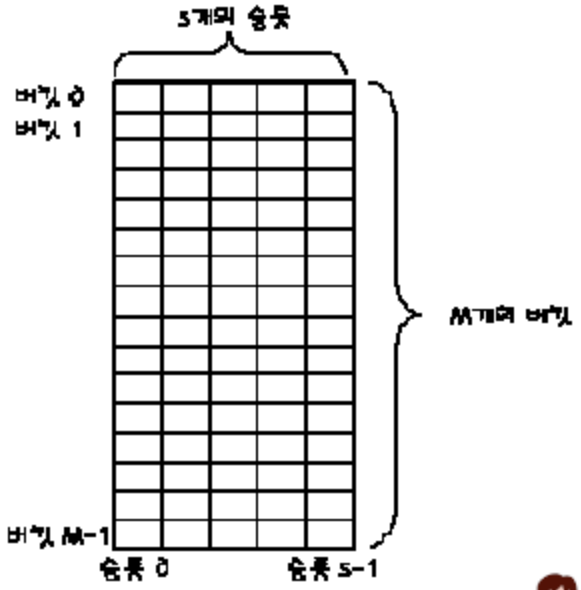
- **해시 함수**(hash function)란 탐색키를 입력으로 받아 **해시 주소**(hash address)를 생성하고 이 해시 주소가 배열로 구현된 **해시 테이블**(hash table)의 인덱스가 된다.
- 예를 들어 영어사전에서는 단어가 탐색키가 되고 이 단어를 해싱 함수를 이용하여 적절한 정수 i로 변환한 다음, 배열 요소 ht[i]에 단어의 정의를 저장하는 것이다.



해시 테이블의 구조

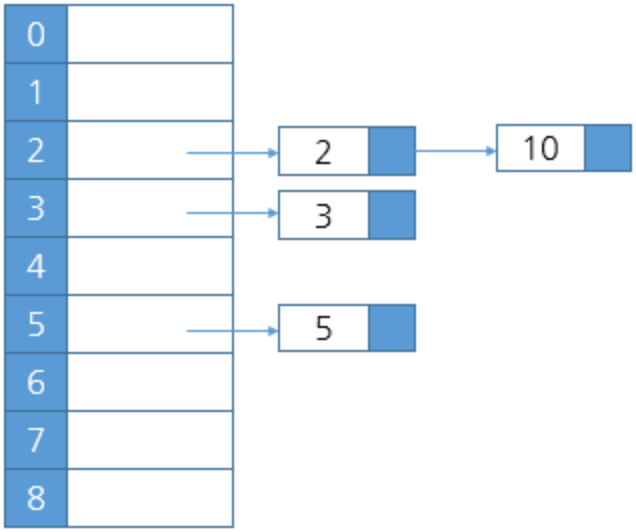
- 해시테이블 ht는 M개의 **버켓**(bucket)으로 이루어지는 테이블로서 ht[0], ht[1], ...,ht[M-1]의 원소를 가진다.
- 하나의 버켓은 s개의 **슬롯**(slot)을 가질 수 있다.

- **충돌**(collision) : 서로 다른 두 개의 탐색키 k1과 k2에 대하여 h(k1) = h(k2)인 경우
- 이러한 충돌이 버켓에 할당된 슬롯 수보다 많이 발생하게 되면 버켓에 더 이상 항목을 저장할 수 없게 되는 **오버플로우**(overflow)가 발생
- 오버플로우를 해결하기 위한 방법이 반드시 필요



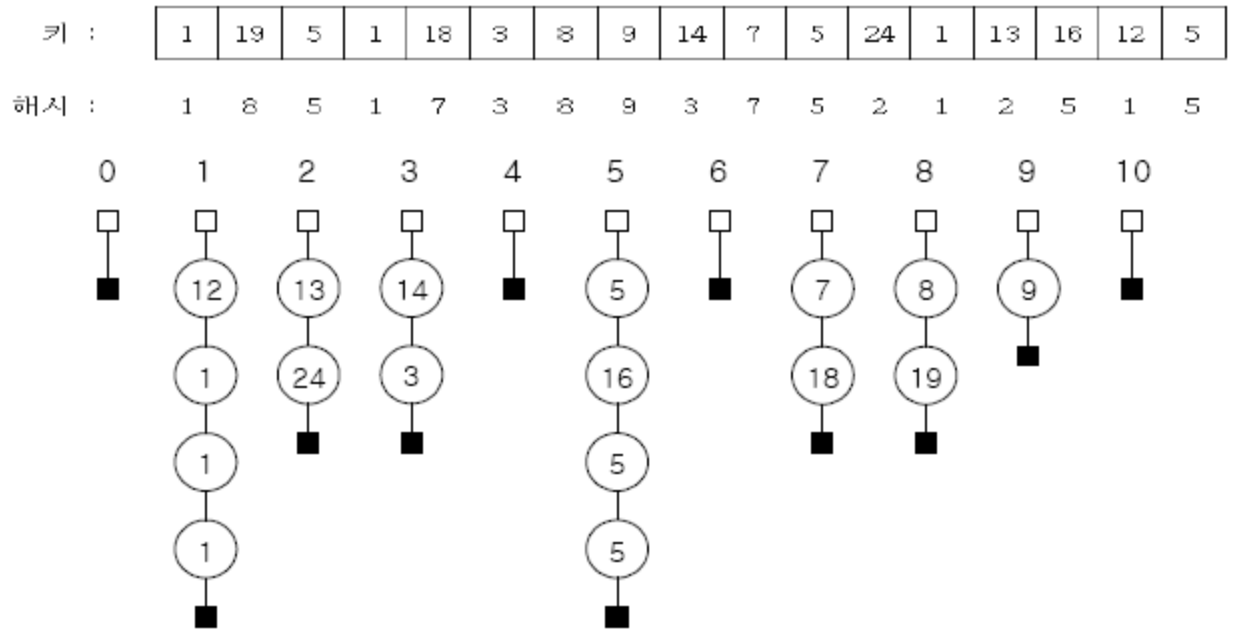
- Chaining: overflow 대처

• 체이닝은 자리에 원소를 저장하는게 아니라 연결 리스트를 저장합니다. 그리고 키가 해당 자리에 들어오려고 할 때 연결리스트에 해당 키를 저장하는 방법입니다. 위에서 발생한 충돌 상황을 체이닝으로 해결한 결과가 아래 그림에 있습니다.



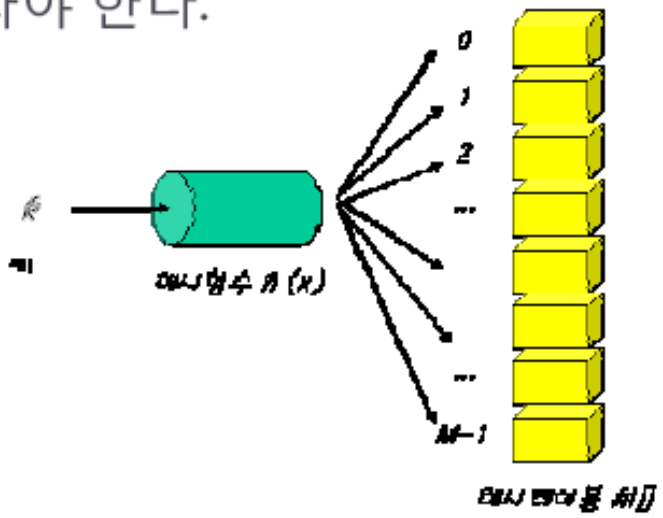
해시 함수의 예

$h(k)$: k mod 11



해시함수

- 좋은 해시 함수의 조건
  - 충돌이 적어야 한다.
  - 해시함수 값이 해시테이블의 주소 영역 내에서 고르게 분포되어야 한다.
  - 계산이 빨라야 한다.



해싱의 특징

- 시간과 공간의 균형
  - 메모리의 제한이 없을 경우 : 키를 메모리 주소로 사용하면 어떤 탐색이든지 한 번의 메모리 접근으로 수행 가능
  - 시간에 제한이 없을 경우 : 최소한의 메모리를 사용하여 데이터를 저장하고 순차 탐색
- 해싱은 이러한 두 극단 사이에서 합리적인 균형을 이룰 수 있는 방법을 제공
- 해싱을 사용하는 목적은 가용한 메모리를 효과적으로 사용하면서 빠르게 메모리에 접근하기 위함

- 이진 트리보다 해싱을 많이 사용하는 이유
  - 간단함
  - 아주 빠른 탐색 시간
- 이진 트리의 장점
  - 동적 (삽입 회수를 미리 알고 있지 않아도 됨)
  - 최악의 경우 성능을 보장 (해싱의 경우 아무리 좋은 해시 함수라도 모든 값을 같은 장소로 해싱하는 경우[Collision] 가 발생)
  - 사용할 수 있는 연산의 종류가 많음 (예 : 정렬 연산)