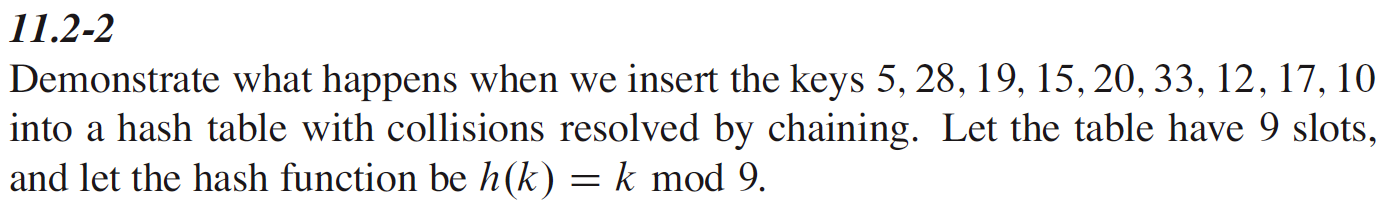
자료구조 알고리즘 Hash 과제 제출

141019 김연우



|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| NIL | 10 | 20 | 12 | NIL | 33 | 15 | NIL | 17 |
|  | 19 | NIL | NIL |  | 5 | NIL |  | NIL |
|  | 28 |  |  |  | NIL |  |  |  |
|  | NIL |  |  |  |  |  |  |  |



* 1. A – B =

1. 임의의 문자열 상수를 x라 하면 x의 임의의 퍼뮤테이션 x’에 대해 h(x) = h(x’)를 만족한다.
   1. 임의의 스트링 x 는 x = 으로 표현된다.
   2. h(x) =
   3. 1번에 의해서 i번째 자리가 섞여도 h(x)값은 일정하다.
   4. 따라서 x의 임의의 퍼뮤테이션 x’에 대해 h(x) = h(x’)를 만족한다.
2. 문자열 비트가 직접 영향을 주지 않는 해쉬함수를 만들 수 있다.
   1. 0~ 1 사이의 소수로 만들어서 m을 곱하는 방식



|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| linear | 22 | 88 |  |  | 4 | 15 | 28 | 17 | 59 | 31 | 10 |
| quadratic | 22 |  | 88 | 17 | 4 |  | 28 | 59 | 15 | 31 | 10 |
| double | 22 |  | 59 | 17 | 4 | 15 | 28 | 88 |  | 31 | 10 |

**C++ Hash**

hash\_map, hash\_set

1. 간략한 설명

C++의 표준 라이브러리인 STL에는 속해있지 않지만, 기본 C++컴파일러에서 대부분 지원하는 Hash 테이블형 자료구조. 새로운 C++표준에는 비슷한 형태의 자료구조인 unordered\_map, unordered\_set이 추가될 것이라고 한다.

1. 언제 사용해야 하는가?
   1. 데이터 개수가 많을 때에도 이론상 검색 속도가 O(1)
   2. 데이터 개수가 적은데 사용하면, 메모리 낭비 & 오버헤드.
   3. 적은 요소를 저장 검색할 때는 list, vector, deque 가 더 좋은 선택일 수 있다.
2. 사용방법

#include <hash\_map>

using namespace stdext;

int main(void)

{

//생성, 할당

hash\_map<Key, Value>\* hashInsetance = new hash\_map < Key, Value > ;

Key \_key = 10;

Value \_value = 4.5f;

pair<Key, Value> \_pair;

auto \_where = hashInsetance->begin();

//insert Data

\_pair = hashInsetance->insert(hash\_map<Key, Value>::value\_type(\_key, \_value));

//erase Data

\_where = hashInsetance->erase(\_where);

//find Data

\_where = hashInsetance->find(\_key);

//lower\_bound

\_where = hashInsetance->lower\_bound(\_key);

//upper\_boudn

\_where = hashInsetance->upper\_bound(\_key);

//reference : http://msdn.microsoft.com/en-us/library/h80zf4bx(VS.80).aspx

}

1. PS
   1. Visual Studio 컴파일러의 hash\_map은 느리다.
   2. 성능이 중요하다면 boost의 unordered\_map을 사용하는 것을 추천
   3. window에서만 사용한다면 마이크로 소프트의 CAtlMap을 사용하는 것을 추천
   4. 참조 : <http://www.hanbit.co.kr/network/view.html?bi_id=1617>