HW10: hashing

제출은 gitLab을 통해서 하며 **http://hconnect.hanyang.ac.kr/2017\_CSE2010\_수업번호/2017\_CSE2010\_수업번호\_학번.git 에 HW10라는 폴더를 만들어 진행.**

\* 프로그램 제출간 유의사항

- 소스코드에는 주석이 있어야 함.

- 주어진 구조체와 input.txt를 사용해야 함

**- 숙제의 소스코드 평가는 linux ubuntu 16.04.2 LTS 버전 gcc 5.4.0에서 함.**

\* 보고서 제출간 유의사항

- 작성한 소스 코드가 첨부되어야 하며, 실행결과가 첨부되어야 함.

- 분량은 제한이 없으나 1~2apge로 간략하게 설명.

- 보고서는 hw10\_학번.확장자(doc, docx, pdf)로 제출  
 >> Merge sort, heap sort 두 토픽의 내용이 들어가야 함.

**제출시간: '17.6.14(23:59) 까지**

**\*지연제출**

- 24시간 이내는 해당 과제 50% 감점, 48시간 이내는 75% 감점.

- 지연제출자는 E-mail(casualab@hanyang.ac.kr)과 gitlab에 모두 제출.

- E-mail제목: "hw10\_학번\_자신의 수업 요일(수, 목)\_이름 " 형식으로 제출.

**hash function / hash\_chain\_find**

해싱이란 데이터를 테이블로 대응시켜 저장하는 방법이다. 해시함수와 체인검색함수를 구현한다. 해시함수(hash\_function)는 해시테이블의 인덱스가 될 해시주소를 생성한다. 체인검색함수(hash\_chain\_find)는 해시테이블에서 키를 탐색한다. 탐색 결과에 따라서 "키를 찾지 못했음", "키를 찾았음" 메세지를 출력한다.

1. Input

주어진 입력 파일(input.txt)에서 입력데이터를 가져오며, 예상출력은 아래와 같다.

Input1.txt

|  |
| --- |
| a b c d e |

예상출력:

|  |
| --- |
|  |

2. hashing ADT

#define KEY\_SIZE 20 // 탐색키의 최대길이

#define TABLE\_SIZE 7 // 해시테이블의 크기

typedef struct

{

char key[KEY\_SIZE];

} element;

struct list

{

element item;

struct list \*link;

};

struct list \*hash\_table[TABLE\_SIZE];

**void hash\_chain\_print(struct list \*ht[])**

- 입력: ht[]는 체인법으로 구현된 해시테이블

- 하는 일: 체인법으로 구현된 해시테이블의 모든 키를 출력한다.

- 출력: void

**int transform(char \*key)**

- 입력: key로 사용할 문자열

- 하는 일: 해시함수를 이용하기 위해 key를 음이 아닌 정수 변환한다. 모든 문자열을 길이에 상관없이 하나의 정수로 만든다. 같은 문자열은 동일한 정수로 변환된다.

- 출력: int형의 문자열을 정수로 변환한 값

**int hash\_function(char \*key)**

- 입력: key로 사용할 문자열

- 하는 일: 문자열을 해시테이블 내의 버킷의 주소(해시주소)로 바꾼다. 해시함수는 중간제곱함수로 만든다. 레코드 키값을 제곱한 후 결과값의 중간에 있는 위치한 3개의 비트를 선택하여 해시 테이블의 홈 주소로 사용한다. 이 때, 키는 정수라고 가정한다. 중간이라 함은 "키 값의 제곱한 자릿수/2"가 된다. 키값의 제곱이 8자리의 2진수가 나온다면 중간은 4번째 부터 위치한 3개의 비트가 된다. 즉, 4,5,6번째의 비트가 테이블의 홈 주소가 된다.

ex) 키값 K를 transform함수에 입력하여 나온 값이 십진수로 14고 해시테이블의 크기가 7일 때 14의 제곱인 196를 2진수로 만들면 110001002이 된다. 이 2진수 8자리 중간(n/2, n/2+1, n/2+2)에 위치한 비트(0012)를 십진수로 취한 1이 return값이 된다.

- 출력: int형의 해시주소

**void hash\_chain\_add(element item, struct list \*ht[])**

- 입력: item은 입력할 해시주소가 들어가 있는 element 구조체

ht[]는 체인법으로 해시주소가 들어가도록 구현된 해시테이블

- 하는 일: 체인법으로 해시테이블 내 item이 적절한 위치에 들어가도록 한다. 이미 등록된 키가 있으면 들어가지 않도록 처리하고 아니라면 체인에 item을 insert한다.

- 출력: void

**void hash\_chain\_find(element item, struct list \*ht[])**

- 입력: item은 해시주소가 들어가 있는 element 구조체

ht[]는 체인법으로 해시주소가 들어가도록 구현된 해시테이블

- 하는 일: 체인검색 함수다. 해시테이블 ht[]에서 item에 들어있는 key를 탐색한다. item에 들어있는 key가 없으면 "키를 찾지 못했음(item.key: %s)\n" 메세지를 콘솔창에 출력하고 있다면 "키를 찾았음(item.key: %s)\n" 메세지를 출력한다.

- 출력: void

3. 프로그램 설명

* 파일명: HW10\_학번.c
* 입력: 주어진 입력 파일(input.txt)의 숫자("1. input"항목을 참고)
* 출력 : 일반 입출력