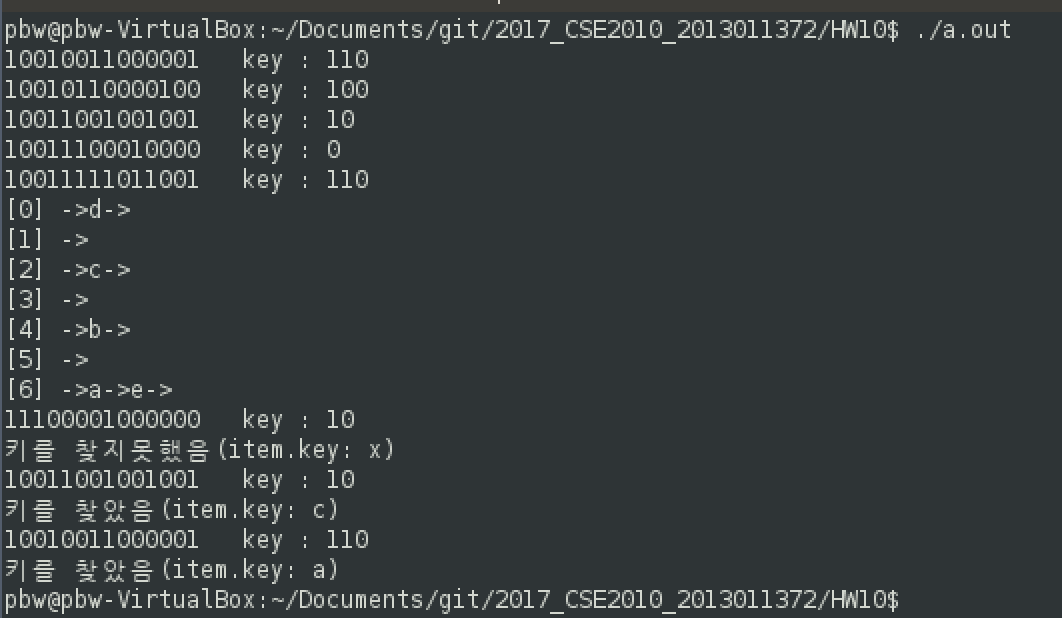
hw10\_2013011372 박병욱

1. 실행결과



2

2. 소스코드

#include <string.h>

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define KEY\_SIZE 20 // 탐색키의 최대길이

#define TABLE\_SIZE 7 // 해싱 테이블의 크기

#define equal(e1,e2) (!strcmp(e1.key,e2.key))

typedef struct

{

char key[KEY\_SIZE];

} element;

struct list

{

element item;

struct list \*link;

};

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

///////숙제로 구현할 코드/////////////

int hash\_function(char \*key)

{

int value = transform(key);

int i ,j; //for find max digit

int mid;

value \*= value;

j=0;

for(i=1; i<value; i \*= 2){

j++;

}

print\_binary(value);

printf("\t key : ");

mid = j/2;

value = value >> mid-2;

value = value & 0x07;

print\_binary(value);

printf("\n");

return value;

}

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

{

//fill in the blank

int hash\_value = hash\_function(item.key);

struct list \*ptr;

struct list \*node\_before=NULL, \*node = ht[hash\_value];

for(; node; node\_before=node, node=node->link){

if(equal(node->item, item)){

printf("키를 찾았음(item.key: %s)\n", item.key);

return;

}

}

printf("키를 찾지못했음(item.key: %s)\n", item.key);

}

void print\_binary(int num){

int t;

if(num <= 1){

printf("%d",num);

return;

}

t = num%2;

print\_binary(num/2);

printf("%d",t);

}

////숙제로 구현할 코드:END////////////

////////실습시간에 다룬 코드(완성된 상태)//////////

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

{

struct list \*node;

int i;

for(i=0;i<TABLE\_SIZE;i++){

printf("[%d] ->", i);

for(node=ht[i]; node; node=node->link){

printf("%s->", node->item.key);

}

printf("\n");

}

}

// 문자로 된 탐색키를 숫자로 변환

int transform(char \*key)

{

int number=0;

// 간단한 덧셈 방식 사용 자연수 생성

while(\*key) number += \*(key++);

return number;

}

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

{

int hash\_value = hash\_function(item.key);

struct list \*ptr;

struct list \*node\_before=NULL, \*node = ht[hash\_value];

for(; node; node\_before=node, node=node->link){

if(equal(node->item, item)){

printf("input err\n");

return;

}

}

ptr = (struct list \*)malloc(sizeof(struct list));

ptr->item = item;

ptr->link = NULL;

if(node\_before) node\_before->link = ptr;

else ht[hash\_value]=ptr;

}

////////실습시간에 다룬 코드(완성된 상태): END//////////

void main()

{

FILE\* f;

element tmp;

char input[KEY\_SIZE]=" ";

f=fopen("input.txt","r");

while(fscanf(f, "%s", &input) != EOF){

strcpy(tmp.key, input);

hash\_chain\_add(tmp, hash\_table);

}

hash\_chain\_print(hash\_table);

strcpy(tmp.key, "x");

hash\_chain\_find(tmp, hash\_table);

strcpy(tmp.key, "c");

hash\_chain\_find(tmp, hash\_table);

strcpy(tmp.key, "a");

hash\_chain\_find(tmp, hash\_table);

}

3. 코드에 대한 설명

hash\_function(), hash\_find()함수를 구현하였고, print\_binary()함수를 새로 만들었음,

print\_binary : 정해진 int타입의 숫자를 2로 나눈값을 재귀하고, 나머지를 재귀후에 프린트한다.

hash\_function(), key를 트랜스폼하여 value를 얻고 value를 제곱한 후 그 밸류2진수의 최대자릿수를 구한다. 그리고 mid를 구하고, mid-2값을 오른쪽으로 쉬프하여 3자리만 읽어온다.

그 후 그 값을 반환한다.

hash\_find() hash\_add()함수와 거의 동일하고, 자리를 찾을경우 키를 찾았다고 프린트한다. 그외에 경우 키를 못찾음