자료구조응용

22. Hashing: Linear Probing, Random Probing

1. **[Linear Probing] 다음과 같은 해시함수와 Linear Probing을 사용하는 해시테이블에 대해 search, insert 함수를 작성하고 그 결과를 출력하는 프로그램을 작성하라.**

<해싱조건>

입력파일(input.txt) :

acos atoi char define exp ceil cos float floor ctime

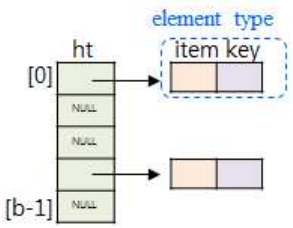
※ 입력문자열의 최대 크기는 10임을 가정한다.

키 변환함수 : 각 입력문자열을 0 이상의 정수로 바꿈 (Program 8.1)
반환된 정수를 해싱함수의 입력 k로 사용

해싱함수(h(k)) : k % b 연산 결과를 반환함

버킷 수 (b) : 11

슬롯 수 (s) : 1



item : 입력문자열
key : 키 변환함수의 반환 값

<실행순서>

① 전역변수로 해시테이블(ht)을 초기화하고, MAX 기호상수를 11로 정의한다.

② 입력파일로부터 읽은 각 문자열과 key값은 element 타입의 구조체를 동적으로 할당받아 저장하고 그 주소를 해당 버킷에 저장한다.

③ 해싱테이블을 최종 생성한 후 인덱스 순서대로 (item, key)를 출력한다.

④ 사용자로부터 문자열을 입력받아 탐색 후 (item, key, 비교횟수)를 출력한다.

<참조 코드>

```
unsigned int stringToInt(char *key)
{/* simple additive approach to create a natural number
    that is within the integer range */
   int number = 0;
   while (*key)
      number += *key++;
   return number;
}
```

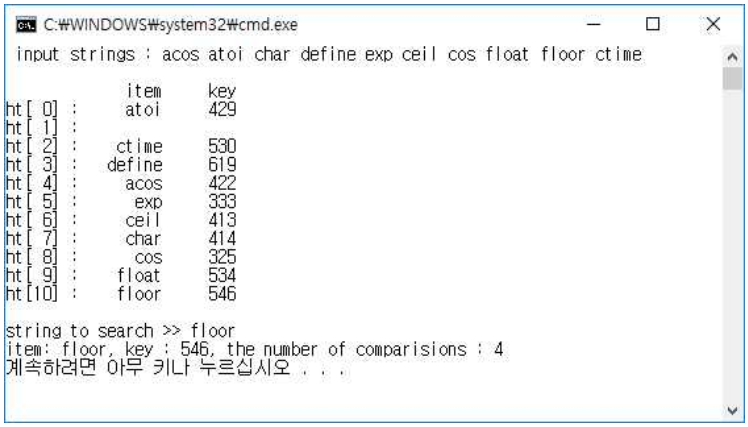
Program 8.1: Converting a string into a non-negative integer

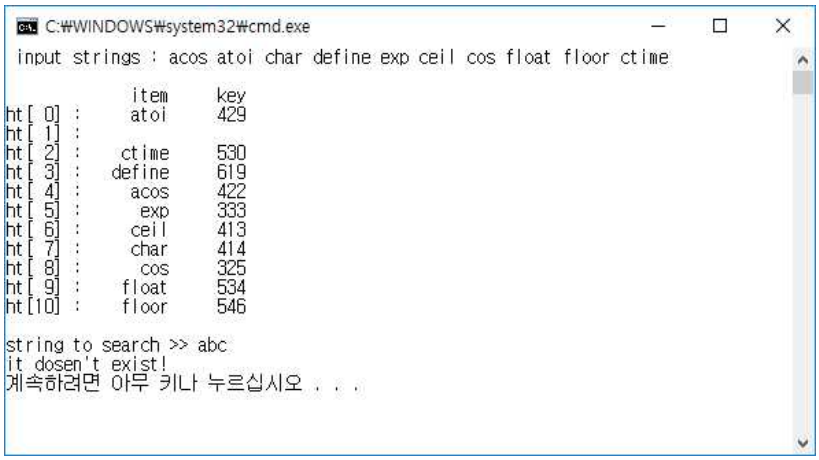
```
element* search(int k)
{/* search the linear probing hash table ht (each bucket has
    exactly one slot) for k, if a pair with key k is found,
    return a pointer to this pair; otherwise, return NULL */
   int homeBucket, currentBucket;
   homeBucket = h(k);
   for (currentBucket = homeBucket; ht[currentBucket]
                && ht[currentBucket]->key != k;) {
      currentBucket = (currentBucket + 1) % b;
                      /* treat the table as circular */
      if (currentBucket == homeBuket)
         return NULL; /* back to start point */
   }
   if (ht[currentBucket]->key == k)
      return ht[currentBucket];
   return NULL;
}
```

Program 8.3: Linear probing

※ 밑줄친 부분 : hash table에 있는 key 항목에 대한 탐색만 고려할 때 OK!
hash table에 없는 key항목에 대한 탐색까지 고려하면 어떻게 수정되어야 할까?
(없는 항목을 참조시 오류가 날수 있음)

<실행결과>


C:₩WINDOWS₩system32₩cmd.exe
input strings : acos atoi char define exp ceil cos float floor ctime
item key
ht[0] : atoi 429
ht[1] :
ht[2] : ctime 530
ht[3] : define 619
ht[4] : acos 422
ht[5] : exp 333
ht[6] : ceil 413
ht[7] : char 414
ht[8] : cos 325
ht[9] : float 534
ht[10] : floor 546
string to search >> floor
item: floor, key : 546, the number of comparisions : 4
계속하려면 아무 키나 누르십시오 . . .


C:₩WINDOWS₩system32₩cmd.exe
input strings : acos atoi char define exp ceil cos float floor ctime
item key
ht[0] : atoi 429
ht[1] :
ht[2] : ctime 530
ht[3] : define 619
ht[4] : acos 422
ht[5] : exp 333
ht[6] : ceil 413
ht[7] : char 414
ht[8] : cos 325
ht[9] : float 534
ht[10] : floor 546
string to search >> abc
it dosen't exist!
계속하려면 아무 키나 누르십시오 . . .

2. [Random Probing] 다음과 같은 division 해시함수와 Random Probing을 사용하는 해시 테이블에 대해 search, insert 함수를 작성하고 그 결과를 출력하는 프로그램을 작성하라.

<해싱조건>

입력파일(input.txt) :

```
5 8 13 7 21 23
```

해싱함수(h(k)): k%b

키 탐색순서 - h(k), (h(k)+s(i))%b, 1≤i≤b-1, s(i)는 유사난수(pseudo random number)

난수생성 : s(i)는 1≤i≤b-1 시퀀스에 대해 1에서 b-1 범위의 난수를 정확하게 한 번씩 생성해야 하며, 매 탐색마다 동일한 seed를 사용하여야 함. 이러한 특징의 난수생성기를 직접 구현하는 대신, C 언어의 srand, rand함수를 활용함

버킷 수 (b) : 8

슬롯 수 (s) : 1

<예>

Input sequence : 5 8 13 7 21 23
Random numbers : 5 2 3 7 1 4 6
Hash table : 8 buckets with 1 slot

	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]
ht	8	23	13		21	5		7

k=5 : h(k) = 5%8 = 5
k=8 : h(k) = 8%8 = 0

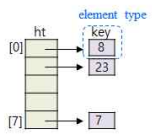
k=13 : h(k) = 13%8 = 5
(h(k)+s(1))%8 = (5+5)%8 = 2

k=7 : h(k) = 7%8 = 7

k=21 : h(k) = 21%8 = 5
(h(k)+s(1))%8 = (5+5)%8 = 2
(h(k)+s(2))%8 = (5+2)%8 = 7
(h(k)+s(3))%8 = (5+3)%8 = 0
(h(k)+s(4))%8 = (5+7)%8 = 4

k=23 : h(k) = 23%8 = 7
(h(k)+s(1))%8 = (7+5)%8 = 4
(h(k)+s(2))%8 = (7+2)%8 = 1

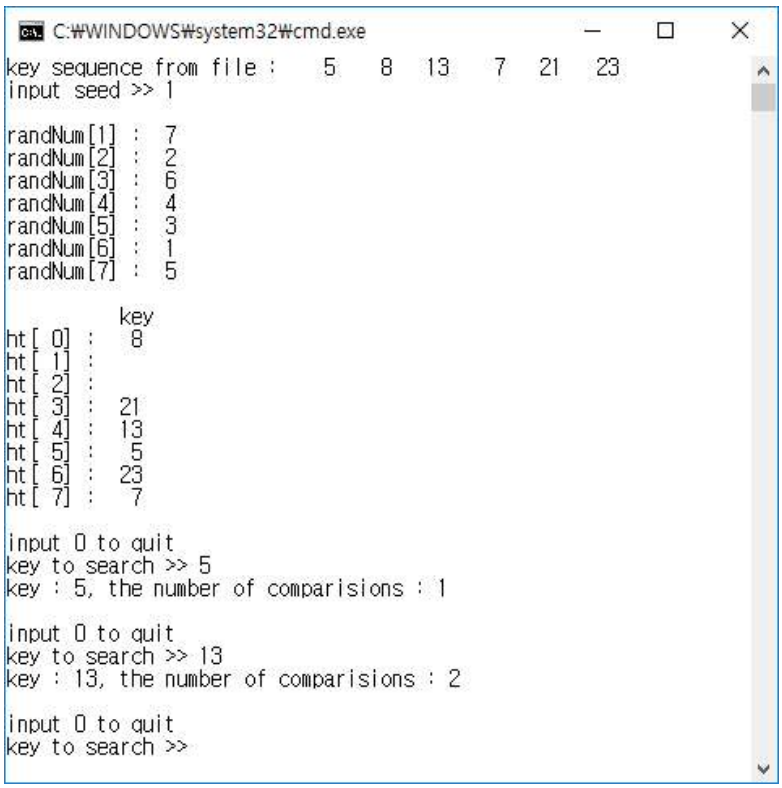
※ 구현



<실행순서>

① 해시테이블(ht)을 element* 타입의 구조체포인터 배열의 전역변수로 선언하고 초기화한다.
※ element는 key 요소만으로 구성된 구조체이다.
※ 입력되는 key 값은 0보다 큰 정수이다.
② 사용자로부터 seed를 입력받는다.
③ 1에서 b-1 범위의 난수 시퀀스를 중복 없이 생성해서 전역변수인 배열에 저장한다.
④ 입력파일로부터 읽은 key값은 element 타입의 구조체를 동적으로 할당받아 저장하고 그 주소를 해당 버킷에 저장한다.
- 해싱테이블(ht)에 더 이상 추가할 수 없을 때는 적절한 메시지를 출력하고 종료한다.
- 중복된 key가 있을 경우에는 적절한 메시지를 출력하고 종료한다.
⑤ 해싱테이블을 최종 생성한 후 인덱스 순서대로 key를 출력하라.
⑥ 사용자로부터 키를 입력받아 탐색 후 (key, 비교횟수)를 출력하되 0이 입력될 때 까지 반복한다.

<실행결과>



3. [Chaining] 다음과 같이 입력파일로부터 문자열을 입력받아 해싱테이블에 추가하는 프로그램을 작성하라.

<해싱조건>

입력파일(input.txt) :

acos atoi char define exp ceil cos float floor ctime

※ 입력문자열의 최대 크기는 10임을 가정한다.

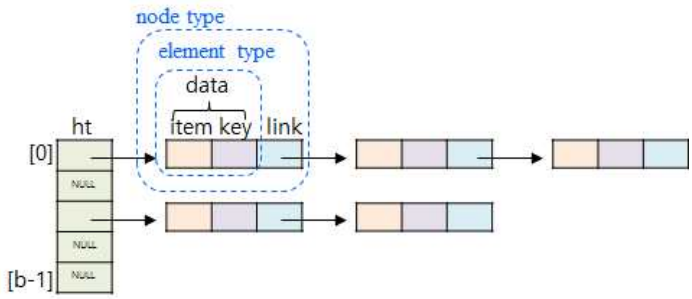
키 변환함수 : 각 입력문자열을 0 이상의 정수로 바꿈 (Program 8.1)
반환된 정수를 해싱함수의 입력 k로 사용

해싱함수(h(k)) : k % b 연산 결과를 반환함

버킷 수 (b) : 11

※각 버킷은 체인으로 구성함

<자료구조>



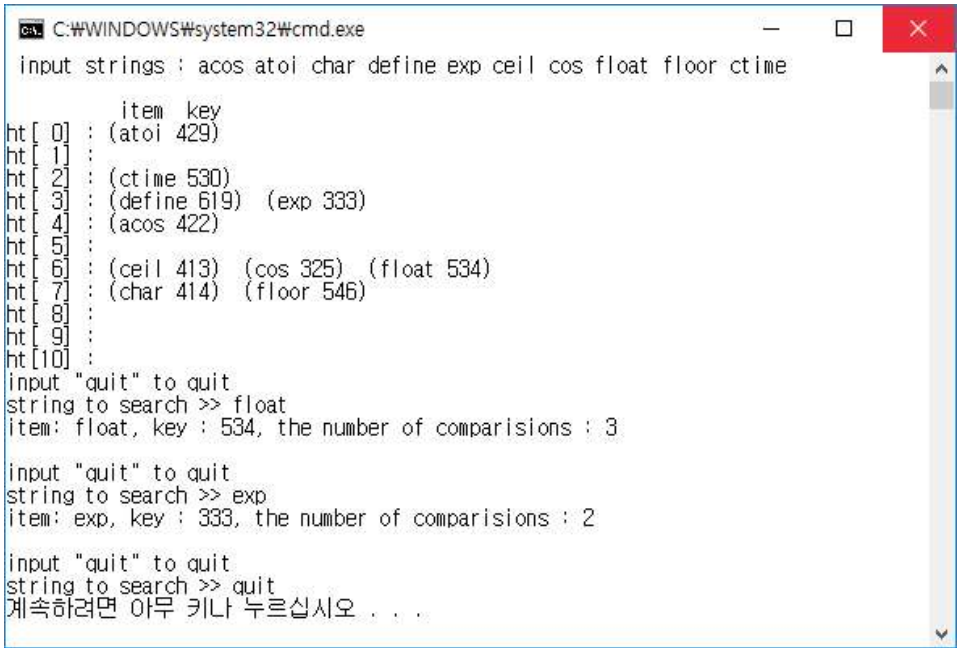
<실행순서>

① 입력파일로부터 읽은 각 문자열과 해당 key값은 node 타입의 구조체를 동적으로 할당받아 저장하고 버킷의 체인에 추가한다. 체인의 각 노드는 element type의 data, nodePointer 타입의 link로 구성되며, element 타입은 key와 item(입력받은 문자열)으로 구성된다.

② 해싱테이블을 최종 생성한 후 인덱스 순서대로 key를 출력하라.

③ 사용자로부터 키를 입력받아 탐색 후 (item, key, 비교횟수)를 출력하되 quit가 입력될 때까지 반복한다.

<실행결과>


C:₩WINDOWS₩system32₩cmd.exe
input strings : acos atoi char define exp ceil cos float floor ctime
item key
ht[0] : (atoi 429)
ht[1] :
ht[2] : (ctime 530)
ht[3] : (define 619) (exp 333)
ht[4] : (acos 422)
ht[5] :
ht[6] : (ceil 413) (cos 325) (float 534)
ht[7] : (char 414) (floor 546)
ht[8] :
ht[9] :
ht[10] :
input "quit" to quit
string to search >> float
item: float, key : 534, the number of comparisions : 3
input "quit" to quit
string to search >> exp
item: exp, key : 333, the number of comparisions : 2
input "quit" to quit
string to search >> quit
계속하려면 아무 키나 누르십시오 . . .

■ 제출 형식

- 공학인증 시스템(ABEEK)에 과제를 올릴 때 제목:
- 1차 제출: 학번 이름 DS-22(1), 2차 제출: 학번 이름 DS-22(2)
- 솔루션 이름 : DS-22
- 프로젝트 이름 : 1, 2, 3
- 실행화면을 캡쳐하여 한글파일에 추가 후 솔루션 폴더에 포함.
- 한글 파일명 : 학번_이름_실습결과.hwp
- 솔루션 폴더를 압축하여 게시판에 제출할 것.
- 압축 파일 명: 학번_이름_DS-22.zip

- 제출은 2회걸쳐 가능(수정 시간 기준으로 처리)