자료구조론

Hashing

```
u@hataeseong-ui-MacBook-Pro:~/Desktop/2017_CSE2010_2016025041/HW10$./a.out
10010011000001    key : 1, 1, 0
10010110000100    key : 1, 0, 0
10011001001001    key : 0, 1, 0
10011100010000    key : 0, 0, 0
10011111011001    key : 1, 1, 0
[0] ->d->
[1] ->
[2] ->c->
[3] ->
[4] ->b->
[5] ->
[6] ->a->e->
11100001000000    key : 0, 1, 0
Can't fine key(item.key: x)
10011001001001    key : 0, 1, 0
Find key(item.key: c)
u@hataeseong-ui-MacBook-Pro:~/Desktop/2017_CSE2010_2016025041/HW10$
```

- 실행 결과 일치

```
int hash_function(char *key)
{
 //fill in the blank
 int arr[20];
 int tmp[20];
 int i, j;
 int int_dec = transform(key); //transform into dec.
 int_dec *= int_dec; // square
 for(i = 1; int_dec > 0; i++){ // transform into binary
   tmp[i] = int_dec % 2;
   int_dec /= 2;
 }
 i--;
```

```
for(j = 1; i > 0; i--, j++){
  arr[j] = tmp[i];
}
i = j;
for(j = 1; j < i; j++){
  printf("%d", arr[j]);
}
// real key
printf("\tkey : %d, %d, %d\n", arr[i / 2], arr[i / 2 + 1], arr[i / 2 + 2]);
return arr[i / 2] * 4 + arr[i / 2 + 1] * 2 + arr[i / 2 + 2];
}
```

- 해싱 함수 : 십진수로 키를 변환한 뒤 제곱을 하고 이진수로 변환을 해서 중간부터 3자리를 real key로 사용

```
void hash_chain_find(element item, struct list *ht[])
{
 //fill in the blank
 hash_function(item.key);
 for(int i = 0; i < 7; i++){
  if(ht[i] == NULL)
   continue;
  if(ht[i]->item.key[0] == item.key[0]){
   printf("Find key(item.key: %s)\n", item.key);
   return ;
  }
 }
 printf("Can't fine key(item.key: %s)\n", item.key);
}
```

- search 함수 : 해싱 함수를 실행한뒤 키가 있으면 Find key~를 출력하고, 없으면 Can't find key~를 출력한다

```
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");
  }
```

```
}
```
- linked list로 구현된 노드를 하나씩 따라가면서 출력

```
// invert search key into int
int transform(char *key)
{
   int number=0;
  // make simple plus natural number
   while(*key) number += *(key++);
   return number;
}
```
- 키를 십진수로 변형

```
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;
    }
  }
```
- 해싱 함수를 실행하고 키가 있으면 input err 출력 없으면 삽입