PA1

2016310932 배현웅

db.h

```
1
2   typedef struct db {
3       /* Hash Table */
4       char *key;
5       int key_len;
6       int value;
7       int val_len;
8       struct db *next;    // store memory in start at db->next
9       /* Something else */
10  } db_t;          // ex) key : EMMA / value : 2
11  db_t *db_open(int size);
12  void db_close(db_t *db);
13  void db_put(db_t *db, char *key, int key_len,
14              char *val, int val_len);
15  /* Returns NULL if not found. A malloc()ed array otherwise.
16   * Stores the length of the array in *vallen */
17  char *db_get(db_t *db, char *key, int key_len,
18              int *val_len);
19
20
21
22
```

2-10 : Struct db설정
Char *key : key 값을 저장하는 문자형 포인터
Key_len : 그 키값의 길이 저장
Value : value 값 저장
Val_len : value 길이 저장
Next – 다음 노드를 지정하는 포인터 설정

db.c (1) – db_open,close

```
25
26  int Lsize;
27  unsigned int hash(char *key,int length, int Lsize){
28      // hash func in here
29      unsigned int sum=0;
30      for(int i=0; i<=length; i++){
31          sum += key[i];
32      }
33      return sum%Lsize;
34  }
35  db_t *db_open(int size)
36  {
37      db_t *db = NULL;
38      db = (db_t*)malloc(sizeof(db_t)*size);
39      Lsize = size;
40      for(int i=0; i<size; i++){ // initializer
41          db->next = NULL;
42      }
43      return db;
44  }
45
46  void db_close(db_t *db)
47  {
48      free(db);
49  }
50
```

26 : Lsize – size 을 전역변수로 설정하기 위해
27-34 : hash 함수 – 엔트리를 128로 하는 해쉬함수 설정
key 가 가지고 있는 문자들을 다 더해서 숫자로 저장.
그 후에 Lsize로 나누는 해쉬함수 정의
35-50 : 데이터베이스 여는 함수 설정
db.h 에서 정의한 db_t 포인터 db를 설정함
Db를 malloc으로 동적할당.
db->next 를 NULL 로 초기화

db.c (2) - db_put

```
51  void db_put(db_t *db, char *key, int key_len,
52              char *val, int val_len) // (char*)&cnt == char* val
53  {
54      // search the key
55      int entry = hash(key,key_len,Lsize);
56      db_t *cur = db+entry;
57      // make new db for key value
58      db_t *new = (db_t*)malloc(sizeof(db_t));
59      new->next = NULL;
60
61      new->key = (char*)malloc(sizeof(char)*(key_len+1));
62      strcpy(new->key,key);
63      new->key_len = key_len+1;
64
65      new->value = *((int*)val);
66      new->val_len = val_len;
67      int val_int = *((int*)val);
68
69      if(val_int == 1){ // if there is (val == 1) add the tail
70          //printf("phase1\n");
71          while(cur->next != NULL){
72              cur = cur->next;
73          }
74          cur->next = new;
75      }  else {   // if not, find the key & insert the val(&cnt)
76          //printf("phase2\n");
77          while(cur->next != NULL){
78              cur = cur->next;
79              if(strcmp(cur->key,key) == 0) {
80                  cur->value = val_int;
81                  break;
82              }
83          }
84          free(new);
85      }
86  }
```

51-86 : DB 에 값 넣어주는 함수
55-56 : 해쉬함수를 이용해 db의 엔트리 위치를 찾는다
58-59 : new라는 db_t 포인터를 위한 동적할당하고 그후에 다음 포인터 NULL 값 처리
61-66 : new 라는 db_t에 key 값과 value 값을 저장한다. Val 가 (char*)val 값으로 넘겨주어서 *((int*)val 을 통해 자료형 변환을 해주었다
67 : db에는 value 값이 정수로 저장되어 있으므로 자료형 변환
69-74 : val == 1일 때(db에 값이 없을 때) 테이블 끝에 붙여준다
75-86 : cur->next 가 NULL,즉 없을 때까지 탐색하며 strcmp(cur->key,key) 비교를 통해 같은 키 값을 찾는다.그 후 cur->value 에 val에 저장된 값을 저장한다.

db.c (3) - db_get

```
89  /* Returns NULL if not found. A malloc()ed array otherwise.
90   * Stores the length of the array in *vallen */
91  char *db_get(db_t *db, char *key, int key_len,
92               int *val_len) // finding if the key in db   // val_len ==
93  {
94      char *value = NULL;
95      int entry = hash(key,key_len,Lsize);
96      db_t *cur = db+entry;   // in db, db[entry] and find it
97
98
99      if(cur->next == NULL){ // nothing in the entry
100         //printf("nothing..\n");
101     } else {
102         //printf("finding...\n");
103         while(cur->next != NULL){
104             cur = cur->next;
105             if(strcmp(cur->key,key) == 0 ){ // find the value
106
107                 value = (char*)malloc(sizeof(char)*(cur->val_len+1));
108                 *(int*)value = cur->value;
109                 val_len = &cur->val_len;
110             } else {
111
112             }
113         }
114     }
115
116     return value;
117 }
118
```

91 :db_get 함수를 통해 value 값을 찾아낸다.
94 : 없으면 NULL 값 리턴하기 위한 초기화
95-96 : 해쉬함수를 통해 db의 엔트리 위치를 찾는다.
99-100 : cur->next 가 NULL 값이라는 건 엔트리위치에 저장된 값이 없다는 의미,즉 return value(=NULL)
101-104 : 엔트리에 저장된 값이 있다면 cur = cur->next 를 통해 하나씩 확인한다.
105 : key 값이 같은 지 확인
107 : value 값을 저장하기 위한 동적할당
108-109 : value 값과 길이를 저장
110-112 : 같지않다면 넘어간다.