```c
/* OpenHash.c */
#include <stdlib.h>
#include <malloc.h>
#include <assert.h>
#include <string.h>
#include "OpenHash.h"

//散列表初始化
void HashNew(HASH *h, int capacity, int keySize, int valSize, HashFunc *hashFn,
HashCmp *cmpFn, HashFree *freeFn)
{
    assert(0 < capacity);
    assert(0 < keySize);
    assert(0 <= valSize);
    assert(NULL != hashFn);
    h->capacity = capacity;
    h->keySize = keySize;
    h->valSize = valSize;
    h->size = 0;
    h->hashFn = hashFn;
    h->cmpFn = cmpFn;
    h->freeFn = freeFn;
    h->listSet = (LIST *)malloc(sizeof(LIST) * capacity);
    assert(h->listSet);
    int i = 0;
    for (; i < capacity; i ++)
    {
        ListNew(&(h->listSet[i]), keySize + valSize, cmpFn, freeFn);
    }
}

//获取散列表数据数量
int HashSize(HASH *h)
{
    return h->size;
}

//获取散列表容量
int HashCapacity(HASH *h)
{
    return h->capacity;
}

//散列表销毁
void HashDispose(HASH *h)
{
    int i = 0;
    for (; i < h->capacity; i ++)
    {
        ListDispose(&(h->listSet[i]));
    }
    free(h->listSet);
    h->listSet = NULL;
    h->size = 0;
}

//散列表读取
void *HashGet(HASH *h, const void *e)
{
    int hashKey = h->hashFn(e);
    return ListSearch(&(h->listSet[hashKey]), e);
}

//散列表插入
int HashPut(HASH *h, const void *e, const void *val)
{
    int hashKey = h->hashFn(e);
    if (NULL != ListSearch(&(h->listSet[hashKey]), e))
    {
        return -1;
    }
    void *data = malloc(h->keySize + h->valSize);
    assert(NULL != data);
```

```
73          memcpy(data, e, h->keySize);
74          memcpy((char *)data + h->keySize, val, h->valSize);
75          if (0 != ListInsert(&(h->listSet[hashKey]), data, 0))
76          {
77              return -1;
78          }
79          free(data);
80          h->size ++;
81          return 0;
82      }
83
84      //散列表删除
85      int HashRemove(HASH *h, void *e)
86      {
87          int hashKey = h->hashFn(e);
88          if (NULL == ListSearch(&(h->listSet[hashKey]), e))
89          {
90              return -1;
91          }
92          if (0 != ListRemove(&(h->listSet[hashKey]), e))
93          {
94              return -1;
95          }
96          h->size --;
97          return 0;
98      }
```