

```

1  /* Avl.h */
2  #ifndef _AVL_H
3  #define _AVL_H
4
5  //0--代表比较相同, >0--代表dataAddr<keyAddr, <0--代表dataAddr>keyAddr
6  typedef int  AvlCmp(const void *keyAddr, const void *dataAddr);
7  typedef void AvlFree(void *);
8  typedef void AvlTraverseOp(void *);
9
10 typedef struct avl_node
11 {
12     struct avl_node *parent;
13     struct avl_node *lc;
14     struct avl_node *rc;
15     int height;
16     char key[0];
17 }AVLNODE;
18
19 typedef struct
20 {
21     AVLNODE *root;
22     AVLNODE *hot;//"命中"节点的父亲
23     int size;
24     int keySize;
25     AvlCmp *cmpFn;
26     AvlFree *freeFn;
27 }AVLTREE;
28
29 //Avl初始化
30 void AvlNew(AVLTREE *avlTree, int keySize, AvlCmp *cmpFn, AvlFree *freeFn);
31 //Avl销毁
32 void AvlDispose(AVLTREE *avlTree);
33 //Avl判空
34 int AvlEmpty(AVLTREE *avlTree);
35 //Avl规模
36 int AvlSize(AVLTREE *avlTree);
37 //Avl树高度
38 int AvlHeight(AVLTREE *avlTree);
39 //Avl先序遍历(非递归)
40 void AvlTravPre(AVLTREE *avlTree, AvlTraverseOp *traverseOpFn);
41 //Avl先序遍历(递归)
42 void AvlTravPreRec(AVLTREE *avlTree, AvlTraverseOp *traverseOpFn);
43 //Avl中序遍历(非递归)
44 void AvlTravIn(AVLTREE *avlTree, AvlTraverseOp *traverseOpFn);
45 //Avl中序遍历(递归)
46 void AvlTravInRec(AVLTREE *avlTree, AvlTraverseOp *traverseOpFn);
47 //Avl后序遍历(非递归)
48 void AvlTravPost(AVLTREE *avlTree, AvlTraverseOp *traverseOpFn);
49 //Avl后序遍历(递归)
50 void AvlTravPostRec(AVLTREE *avlTree, AvlTraverseOp *traverseOpFn);
51 //Avl层序遍历
52 void AvlTravLevel(AVLTREE *avlTree, AvlTraverseOp *traverseOpFn);
53 //Avl中查找关键码所在节点
54 AVLNODE *AvlSearch(AVLTREE *avlTree, const void *e);
55 //Avl中插入关键码
56 AVLNODE *AvlInsert(AVLTREE *avlTree, const void *e);
57 //Avl中删除关键码所在节点, 返回值: 0成功, !0失败
58 int AvlRemove(AVLTREE *avlTree, void *e);
59 //Avl中删除关键码所在节点(无需深度删除关键码), 返回值: 0成功, !0失败
60 int AvlRemoveU(AVLTREE *avlTree, void *e);
61 #endif

```