哈希查找过程

哈希表查找

在哈希表上查找的过程和哈希造表的构造过程基本一致。

1）给定K值，根据构造表时所用的哈希函数求哈希地址j，

2）若此位置无记录，则查找不成功；

否则比较关键字，若和给定的关键字相等则成功；

否则根据构造表时设定的冲突处理的方法计算“下一地址”， 重复2)

1. 哈希表查找算法

```
Status SearchHash(HashTable H, KeyType key, int &p, int &c){
/*在开放定址哈希表H中查找关键字为key的数据*/
/*用c记录发生冲突的次数，初值为0*/
p=Hash(k); /*求哈希地址*/
while(H.data[p].key!=NULL && H.data[p].key!=key)
 /*该位置填有数据且与所查关键字不同*/
   collision(p, ++c); /*求下一探查地址p*/
if(H.data[p].key==key )
   return SUCCESS;   /*查找成功,p返回待查数据元素位置*/
else return UNSUCCESS; /*查找不成功,p返回插入位置*/
}
```

2. 哈希表插入算法

```
Status InsertHash(HashTable &H, DataType e){
/*查找不成功时在H中插入数据元素e，并返回SUCCESS*/
/*若冲突次数过大，则重建哈希表*/
c=0;
if(SearchHash (H, e.key, p, c))
   return UNSUCCESS; /*数据已在哈希表中，不需插入*/
else if(c<hashsize[H.sizeindex]/2){
          H.data[p]=e; ++H.count; /*次数c还未达到上限，插入e*/
          return SUCCESS;
        }
      else {
         RecreatHashTable(H);  /*重建哈希表*/
         return SUCCESS;
      }
}
```

哈希表查找与插入算法举例

- 关键字序列为：
- {19，14，23，01，68，20，84，27，55，11，10，79}
- 哈希函数为H（key）= key mod 13
- 采用线性探测处理冲突

建立哈希查找表如下：请查找关键字为84的记录

0	1	2	3	4	5	6	7	8	9	10	11	12
	14	01	68	27	55	19	20	84	79	23	11	10

Key=84
哈希地址H(84)=6，因为e.data[6]不空，且e.data[6].key=19≠84，冲突
冲突处理H1=(6+1)MOD13=7，e.data[7]不空，且e.data[7].key=20≠84, 冲突
冲突处理H2=(6+2)MOD13=8，e.data[8]不空，且e.elem[8].key=84，查找成功，返回数据在哈希表中的序号8。

哈希表查找与插入算法举例

- 关键字序列为：
- {19，14，23，01，68，20，84，27，55，11，10，79}
- 哈希函数为H（key）= key mod 13
- 采用线性探测处理冲突

建立哈希查找表如下：请查找关键字为38的记录

0	1	2	3	4	5	6	7	8	9	10	11	12
	14	01	68	27	55	19	20	84	79	23	11	10

Key=38

哈希地址H(38)=12，因为e.data[12]不空，且e.data[12].key=10≠38, 冲突

冲突处理H1=(12+1)MOD13=0，由于e.data[0]没有存放数据，表明哈希表中不存在关键字为38的记录，查找失败。