静态查找表

```
ADT StaticSearchTable {
    /**
    * 功能：构造一个含n个数据元素的静态查找表st
    * 参数：st 静态查找表；n 元素个数；
    */
    Create(st, n);

    /**
    * 功能：销毁表st
    * 参数：st 静态查找表；
    */
    Destory(st);

    /**
    * 功能：若静态查找表st中存在其关键字
    * 等于key的数据元素，则函数值为该元素
    * 的值或在该表中的位置，否则为“空”
    * 参数：st 静态查找表；key 关键字；
    */
    Search(st, key);

    /**
    * 功能：按照某种次序对静态查找表st的每个元素调用函数
    * visit一次且仅一次
    * 参数：st 静态查找表，visit() 访问函数指针
    */
    Traverse(st, visit());
};
```

顺序表的查找

哨兵查找与普通查找相比较，牺牲了一个存储空间

在查找之前先对顺序表下标为0的关键字赋值，这样必然会找到元素，目的在于免去查找过程中每一步都要检测整个表是否查找完毕。在此，下标为0的元素起到了监视哨的作用

这只是一个程序设计技巧的改进，然而实践证明，这个改进能使顺序查找在长度大于等于1000时，进行一次查找的平均时间几乎减少一半

当然，监视哨也可以设置在高下标处

性能分析

通常以“其关键字和给定值进行过比较的记录个数的平均值”作为衡量查找算法好坏的依据

定义：为确定记录在查找表中的位置，需和给定值进行比较的关键字个数的期望值称为查找算法在查找成功时的平均查找长度（Average Search Length）

平均查找长度

对于含有n个记录的表，查找成功的平均查找长度为

- ASL = from (i = 1) to (n) Pi * Ci

Pi为查找表中第i个记录的概率

Ci为找到表中其关键字与给定值相等的第i条记录时，和给定值已进行过比较的关键字个数

从顺序查找的过程可见，Ci取决于所查记录在表中的位置；查找表中最后一个记录时，仅需要比较1次；而查找表中第一个记录时，需要比较n次；一般情况下，Ci等于n - i + 1

则在等概率情况下顺序查找的平均查找长度为

- ASL = from (i = 1) to (n) n - i + 1
- = (n + 1) / 2

因此，对记录的查找概率不等的查找表，应该提前按照概率大小升序排序，以便提高查找概率

优缺点

顺序查找的缺点是平均查找长度较大，随着n越大，查找效率越低；

最大的优点是：算法简单且适应面广。对结构没有任何要求，无论记录是否按关键字有序均可应用

有序表的查找

折半查找（Binary Search）的查找过程是：先确定待查记录所在的范围（区间），然后逐步缩小范围直到找到或找不到该记录为止

折半查找过程是以处于区间中间位置地关键字和给定值进行比较，若相等则查找成功；若不相等则缩小查找范围，直到新的中间位置记录地关键字等于给定值；或者查找区间的大小小于0时（说明查找失败）为止

性能分析

折半查找在查找不成功时和给定值进行比较的关键字个数最多也不会超过 floor(log2(n)) + 1

折半查找的平均查找长度为

- ASL = (n + 1) / n * (log2(n+1)) - 1

折半查找的效率比顺序查找要高，但是折半查找仅仅适用于有序表，且限于顺序存储结构，对线性链表也无法有效地进行折半查找

索引顺序表的查找

若以索引顺序表标识静态查找表，则Search函数可用分块查找来实现

分块查找又称索引顺序查找，这是顺序查找的一种改进方法；在此查找法中，除表本身以外，尚需建立一个“索引表”

索引表

例如，表中含有18条记录，可分成三个子表，对每个子表建立一个索引项，其中包含：关键字项（其值为该子表内最大的关键字）和指针项（指示该子表第一个记录在表中的位置）

索引表按关键字有序，则表有序或者分块有序（后子表所有记录的关键字均大于前子表中的最大关键字）

分块查找的步骤

因此，分块查找需要分两步进行：先确定待查记录所在的块（子表），然后在块中顺序查找

由于索引项组成的索引表按关键字有序，则确定块的查找可以用顺序查找，也可以用折半查找；而块中记录是任意排序的，则在块中只能是顺序查找；因此分块查找的算法是这两种算法的简单合成

分块查找的平均查找长度

1. ASL = Lb + Lw

- Lb为查找索引表确定所在块的平均查找长度
- Lw为在块中查找元素的平均查找长度

2. 若用顺序查找确定所在块，则分块查找的平均查找长度为

- ASL = (1 / 2) * ((n / s) + s) + 1

3. 若用折半查找确定所在块，则分块查找的平均查找长度为

- ASL = log2((n / s) + 1) + (s / 2)