数据结构&查找

回顾

- 算法
- 时间复杂度
- 数据结构
  - 数组
  - 链表
  - 栈&递归
  - 队列

数据结构

散列表/哈希表

散列表/哈希表（非线性）

- 是根据关键字和值 (key & value) 直接访问内存存储位置的数据结构
- 记录的存储位置=f(key)，这里的对应关系f就称为散列函数，又称为哈希 (hash)函数

散列表/哈希表

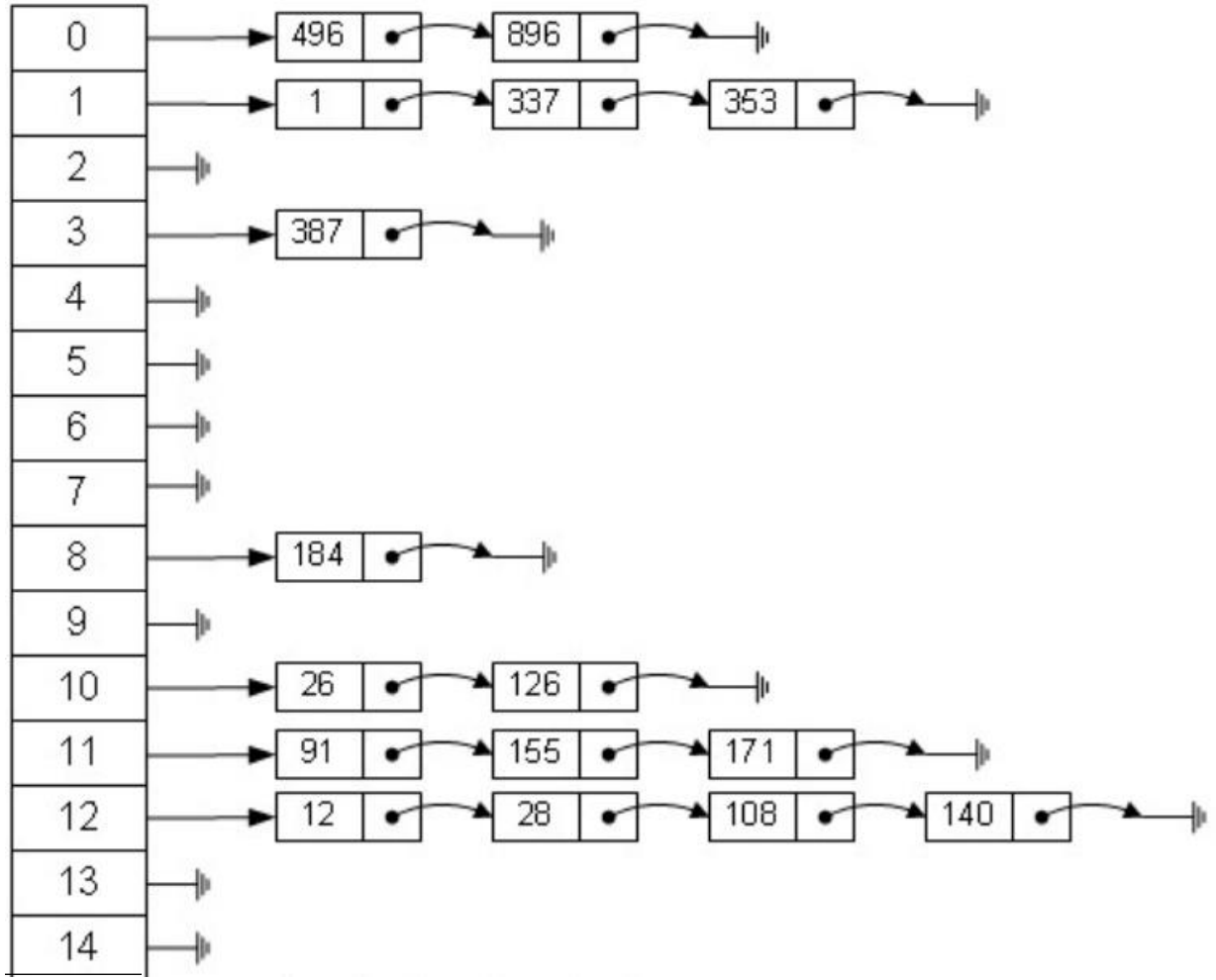
- 哈希函数构造方法（直接寻址/平方取中/除留余数/数字分析等）
- 哈希冲突(Collision)：在构造哈希表时，对于两个不同的关键字，通过哈希函数计算哈希地址后得到了相同的哈希地址。
- 均匀散列函数(Uniform Hash function)：对于关键字集合中的任一个关键字，经散列函数映象到地址集合中任何一个地址的概率是相等的。（使关键字经过散列函数得到一个“随机的”地址，从而减少冲突）

哈希函数构造方法（平方取中）

- 例：我们把英文字母在字母表中的位置序号作为该英文字母的内部编码。例如K的内部编码为11，E的内部编码为05，Y的内部编码为25，A的内部编码为01, B的内部编码为02。由此组成关键字“KEYA”的内部代码为11052501，同理我们可以得到关键字“KYAB”、“AKEY”、“BKEY”的内部编码。之后对关键字进行平方运算后，取出第7到第9位作为该关键字哈希地址，如下图所示

关键字	内部编码	内部编码的平方值	H(k)关键字的哈希地址
KEYA	11052501	122157778355001	778
KYAB	11250102	126564795010404	795
AKEY	01110525	001233265775625	265
BKEY	02110525	004454315775625	315

散列表/哈希表-链表法/拉链法



散列表/哈希表

- 特点
  - 访问速度快
  - 需要额外的内存空间
  - 无序
  - 可能会有哈希冲突
- 适用场景（适合无序、需要快速访问的情况）
  - 缓存
  - 快速查找
- 作业：分析自己在入门考核中提交的哈希函数/当时写不会的重新写一个

查找

顺序查找 二分查找 哈希查找

查找

- 在一些（有序的/无序的）数据元素中，通过一定的方法找出与给定关键字相同的数据元素的过程叫做查找。也就是根据给定的某个值，在查找表中确定一个关键字等于给定值的记录或数据元素。
- 查找算法分类：
  - 静态查找和动态查找；
  - 无序查找和有序查找。
    - 无序查找：被查找数列有序无序均可；
    - 有序查找：被查找数列必须为有序数列。
- 平均查找长度（Average Search Length，ASL）

顺序查找

- 顺序查找/线形查找是无序查找。
- 从数据结构线形表的一端开始，顺序扫描，依次将扫描到的结点关键字与给定值key相比较，若相等则表示查找成功（e.g.返回查找值所在的位置）；若扫描结束仍没有找到关键字等于k的结点，表示查找失败（e.g.返回-1）。
- 时间复杂度：O(n)
- 适用于存储结构为顺序存储或链接存储的线性表。

二分查找

- 二分查找/折半查找是有序查找。
- 用给定值k先与中间结点的关键字比较，中间结点把线形表分成两个子表，若相等则查找成功；若不相等，再根据k与该中间结点关键字的比较结果确定下一步查找哪个子表，这样递归进行，直到查找到或查找结束发现表中没有这样的结点。
- mid = (low + high)/2
- 时间复杂度：$O(\log_2 n)$
- 只适用于元素已经是有序排列好的情况，如果是无序的则要先进行排序操作。
- 作业：代码实现

三分/四分/……？

mid = (low + high)/2

→mid = low + 1/2*(high - low)

→mid = low + (key - a[low])/(a[high] - a[low])*(high - low)

$$mid = \frac{low + high}{2} = low + \frac{1}{2}(high - low)$$ 改成 $$mid = low + \frac{key - a[low]}{a[high] - a[low]}(high - low)$$

key = 1
mid = 0+(1-1)*(99-0)/(100-1)
= 0+0*99/99
= 0

key = 100
mid = 0+(100-1)*(99-0)/(100-1)
=0+99*99/99
=0+99
=99

插值查找

- 插值查找是有序查找。
- 插值查找是根据查找关键字与查找表中最大最小记录关键字比较后的查找方法。插值查找基于二分查找，将查找点的选择改进为自适应选择，提高查找效率。
- 适用于数据量较大&关键字值分布均匀的集合。（关键字分布不均匀的情况下，该方法不一定比二分查找效率高）
- 时间复杂度：$O(\log_2(\log_2 n))$
- 作业：代码实现

哈希查找

- 哈希查找的操作步骤：
  (1)用给定的哈希函数构造哈希表；
  (2)根据选择的冲突处理方法解决地址冲突；
  (3)在哈希表的基础上执行哈希查找。
- 单纯论查找的时间复杂度：O(1)
- 适用于：无冲突的Hash表

参考资料

- https://baike.baidu.com/item/%E5%93%88%E5%B8%8C%E8%A1%A8/5981869
- https://blog.csdn.net/xiaodongdonglht/article/details/95044327