哈希表

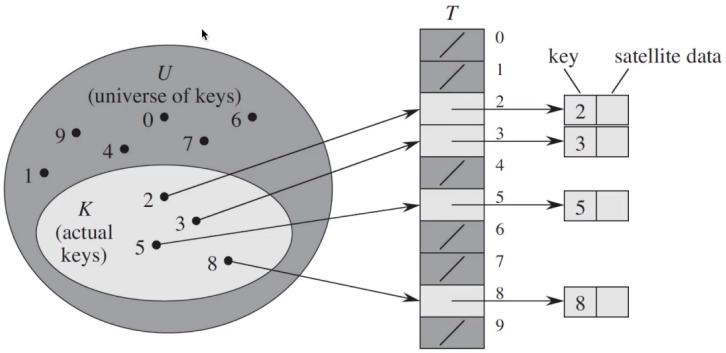
https://www.bilibili.com/video/BV1tq4y1376m?p=62

字典和集合都是使用哈希表实现的

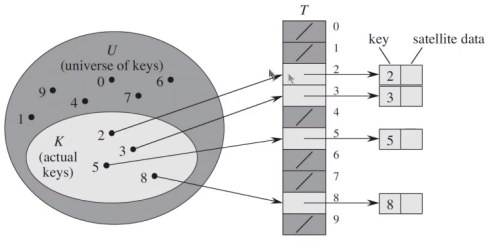
- 哈希表一个通过哈希函数来计算数据存 储位置的数据结构，通常支持如下操作：
  - insert(key, value)：插入键值对(key,value)
  - get(key)：如果存在键为key的键值对则返回其value，否则返回空值
  - delete(key)：删除键为key的键值对

直接寻址表

- 当关键字的全域U比较小时，直接寻址是一种简单而有效的方法。



- 直接寻址技术缺点：
  - 当域U很大时，需要消耗大量内存，很不实际
  - 如果域U很大而实际出现的key很少，则大量空间被浪费
  - 无法处理关键字不是数字的情况



复杂度O（1）

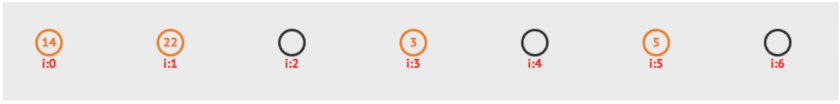
哈希

直接寻址表+哈希=哈希表

- 直接寻址表：key为k的元素放到k位置上
- 改进直接寻址表：哈希（Hashing）
  - 构建大小为m的寻址表T
  - key为k的元素放到h(k)位置上
  - h(k)是一个函数，其将域U映射到表T[0,1,...,m−1]

哈希表

- 哈希表（Hash Table，又称为散列表），是一种线性表的存储结构。哈希表由一个**直接寻址表**和一个**哈希函数**组成。哈希函数h(k)将元素关键字k作为自变量，返回元素的存储下标。
- 假设有一个长度为7的哈希表，哈希函数h(k)=k%7。元素集合{14,22,3,5}的存储方式如下图。



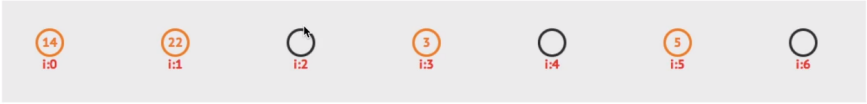
哈希冲突

- 由于哈希表的大小是有限的，而要存储的值的总数量是无限的，因此对于任何哈希函数，都会出现两个不同元素映射到同一个位置上的情况，这种情况叫做哈希冲突。
- 比如h(k)=k%7, h(0)=h(7)=h(14)=...

解决哈希冲突

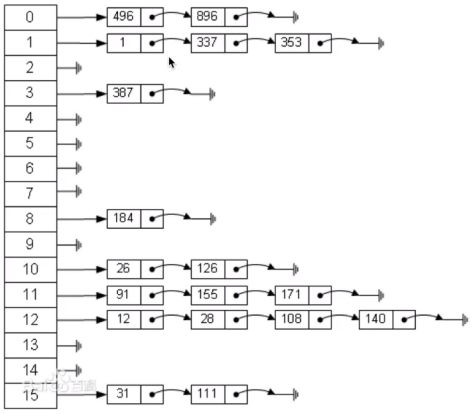
开发寻址法

- 开放寻址法：如果哈希函数返回的位置已经有值，则可以向后探查新的位置来存储这个值。
  - 线性探查：如果位置i被占用，则探查i+1, i+2,……
  - 二次探查：如果位置i被占用，则探查$i+1^2$,$i-1^2$,$i+2^2$,$i-2^2$,……
  - 二度哈希：有n个哈希函数，当使用第1个哈希函数h1发生冲突时，则尝试使用h2,h3,……



拉链法

- 拉链法：哈希表每个位置都连接一个链表，当冲突发生时，冲突的元素将被加到该位置链表的最后。



常见哈希函数

- 除法哈希法：
  - h(k) = k % m
- 乘法哈希法：
  - h(k) = floor(m*(A*key%1))
- 全域哈希法：
  - $h_{a,b}(k)$ = ((a*key + b) mod p) mod m　a,b=1,2,...,p−1

哈希表的应用

集合和字典

- 字典与集合都是通过哈希表来实现的。
  - a = {'name': 'Alex', 'age': 18, 'gender': 'Man'}
- 使用哈希表存储字典，通过哈希函数将字典的键映射为下标。假设h('name') = 3, h('age') = 1, h('gender') = 4，则哈希表存储为[None, 18, None, 'Alex', 'Man']
- 如果发生哈希冲突，则通过拉链法或开发寻址法解决

md5算法

- MD5(Message-Digest Algorithm 5)**曾经**是密码学中常用的哈希函数，可以把任意长度的数据映射为128 位的哈希值，其曾经包含如下特征:
- 1. 同样的消息，其MD5值必定相同;
- 2. 可以快速计算出任意给定消息的MD5值;
- 3. 除非暴力的枚举所有可能的消息，否则不可能从哈 希值反推出消息本身;
- 4. 两条消息之间即使只有微小的差别，其对应的MD5 值也应该是完全不同、完全不相关的;
- 5. 不能在有意义的时间内人工的构造两个不同的消息 使其具有相同的MD5值。

- 应用举例：文件的哈希值
  - 算出文件的哈希值，若两个文件的哈希值相同，则可认为这两个文件是相同的。因此：
    - 1. 用户可以利用它来验证下载的文件是否完整。
    - 2. 云存储服务商可以利用它来判断用户要上传的文件 是否已经存在于服务器上，从而实现秒传的功能， 同时避免存储过多相同的文件副本。

SHA2算法

- 历史上MD5和SHA-1曾经是使用最为广泛的 cryptographic hash function，但是随着密码学的发展，这两个哈希函数的安全性相继受到了各种挑战 。
- 因此现在安全性较重要的场合推荐使用SHA-2等新的更安全的哈希函数。
- SHA-2包含了一系列的哈希函数: SHA-224, SHA- 256, SHA-384, SHA-512, SHA-512/224, SHA- 512/256，其对应的哈希值长度分别为224, 256, 384 or 512 位。
- SHA-2具有和MD5类似的性质(参见MD5算法的特征)。

- 应用举例：
- 例如，在比特币系统中，所有参与者需要共同解决如下问 题:对于一个给定的字符串U，给定的目标哈希值H，需 要计算出一个字符串V，使得U+V的哈希值与H的差小于 一个给定值D。此时，只能通过暴力枚举V来进行猜测。 首先计算出结果的人可获得一定奖金。而某人首先计算成 功的概率与其拥有的计算量成正比，所以其获得的奖金的 期望值与其拥有的计算量成正比。

哈希算法不能反解

python里的哈希库里有