`

图解算法四

散列表——最有用的基本数据结构之一

散列表内部机制：

- 实现
- 冲突
- 散列函数

注：其中最重要的要理解散列函数的机制。

散列函数

- 散列函数是什么样的函数？即，无论给它一个什么样的数据，它都会返回一个数字。
- 散列函数输出的数字必须一致。
    - 即：输入apple，假如返回2，则每次输入apple都将返回2，否则它将毫无意义。
- 它必须对不同的输入返回不同的数字。
    - 即：输入apple，假如返回2，则输入potato将返回3，否则它也将毫无意义。
- 散列函数总是将同样的输入映射到相同的索引。
- 散列函数总是将不同的输入映射到不同的索引。
- 散列函数总是知道数组的长度，从而只返回有效的索引。

注：数组和链表都被注解映射到内存，但散列表更复杂，它使用散列函数来确定元素的存储位置。

有关散列表的应用场景

- 散列表：也被称呼为散列映射、映射、字典和关联数组。
    - 它的速度很快 O(1)，散列表使用数组存储数据，因此获取元素和数组速度相同。
    - 散列表由键和值组成，散列表将键映射到值。
- 散列表多用于数据量大时的查找应用场景：
    - 比如对网站的访问时。
        - 比如访问我的网站，当输入`https://www.xipengheng.cn`时
        - DNS服务就会解析为IP地址为->102.14.xx.13x
        - 这是将网址映射到IP地址，散列表是提供这种功能的方式之一
    - 比如用于防止重复
        - 类似投票功能，检查当前账户是否参与过投票
        - 因为要检查当前用户是否存在投票列表中，存在则拒绝投票
        - 此时的记录名单会非常长，所以要用散列表来记录比对
    - 将散列表用于缓存
        - 缓存的工作原理：网站将数据记录，不再重新计算
        - 因为向服务请求，服务器会做些处理，然后生成网页返回
        - 比如购物网站，经常进行登录，注销操作显示主页等
        - 那么，它可以让服务器不再生成这些，而把主页存储起来，需要时直接发送用户即可
        - （将常用页面URL映射到页面数据）

    - 优点：提升用户体验，更快查看到网页；降低服务器负担，使它做的工作减少。

注：有关DNS解析相关在我的这篇博文里https://www.xipengheng.cn/?p=1019

注：缓存是一种常用的加速方式，所有大型网站都使用缓存，而缓存的数据就存储在散列表中！

有关散列表中的冲突

- **散列表不会也不可能将不同的键映射到数组的不同位置**
- 当不存在冲突时，散列表的查询时间复杂度一般为O(1)
- 场景一：
    - 假设我拥有一个数组，它包含10个位置，而我有11个数组，这时第11个数据又分配到第一个位置
    - 第1个和第11数据都分配到了第一个位置，这种情况称之为**冲突**：两个键分配相同位置
    - 因为后者将覆盖第一次存储的数据，为了避免冲突，解决方案如下：
        - 当两个键映射到同一位置时，就在此位置存储一个链表。
        - 此时访问第二三四速度依然很快，而访问第一个位置时要稍慢些。
        - 当新增存储的链表很短时，这种情况无关紧要。
        - 若排序时，大量冗余数据都排在第一位置，则会很糟糕，散列表速度会很慢
            - 解决方案：避免散列函数将大量键映射到同一位置
            - 最理想的情况是，散列函数将键均匀的映射到散列表的不同位置

注：因为链表存入快，但读取很慢，因此要避免散列表中出现较长的链表。

有关散列表中影响性能的因素

- 我们已知以下条件

方式	查找时间
简单查找	O(n)
二分查找	O(logn)
散列表查找	O(1)

- 那么所有的操作呢?

操作	散列表（平均）	散列表（最糟）	数组	链表
查找	O(1)	O(n)	O(1)	O(n)
插入	O(1)	O(n)	O(n)	O(1)
删除	O(1)	O(n)	O(n)	O(1)

- 以上可知，散列表的插入删除速度与链表一样快，而查找的速度与数组一样快，兼具两者的优点
- 但在最糟的情况下（冲突时），散列表的各种操作都会变得很慢
- 提升它的性能就在于如何避免最糟情况，即：避免冲突
    - 较低的装填因子
    - 良好的散列函数

什么是散列表中的装填因子?

- 装填因子很容易计算，如下所示：

$$\frac{\text{散列表包含的元素数}}{\text{位置总数}}=\text{装填因子}$$

- 比如右侧数组五个位置，有两个填充 `| -- | a | -- | -- | b |` 装填因子为0.4
- 若装填因子大于1，比如上方数组中有10个数据，那么以为数量超过数组位置数，则会产生链表。
- 为避免这种情况，需要再散列表中添加位置称之为：**调整长度**

注：常规的操作是，当装填因此大于0.7时，就应该进行扩充了。

什么是良好的散列函数

- 良好的散列函数使数组中的值均匀分布
- 糟糕的散列函数使值扎堆，导致大量的冲突产生

注：由于正常情况下接触不到散列函数这里不赘述，有兴趣可以看下SHA函数。

总结

> **由以上可知：**

- 可根据散列函数和数组创建散列表
- 冲突很糟糕，会极大程度影响散列表性能，应尽量避免
- 散列表的查找、删除和插入都非常快
- 若散列表中装填因子大于0.7就应该考虑调整散列表长度了
- 散列表适用于防重复（经常查找）
- 散列表也常用于对数据的缓存（比如在对web服务器的数据请求上）