散列表概念：

- 使用键--通过散列技术---找到对应值的位置
- 散列技术就是将记录的位置与它的键建立唯一关系
  - 存储位置 = f（关键字）---这个f函数成为散列函数或者叫哈希函数
- 采用散列技术将记录存在在一块连续的存储空间中，这块连续存储空间称为散列表或哈希表

散列函数构造方法：

最常用：除留余数法

- **mod是取模（求余数）**的意思。事实上，这方法不仅可以对关键字直接取模，也可以再折叠、平方取中后再取模。
- 若散列表的表长为m，通常p为小于或等于表长（最好接近m）的最小质数或不包含小于20质因子的合数。

平方取中法：

- 假设关键字是1234，那么它的平方就是1522756，再抽取中间的3位就是227，用做散列地址。
- 平方取中法比较适合不知道关键字的分布，而位数又不是很大的情况。

开放定址法

折叠法

- 折叠法是将关键字**从左到右分割成位数相等的几部分**（注意最后一部分位数不够时可以短些），然后将这几部分叠加求和，并按散列表表长，取后几位作为散列地址。
- 比如关键字是9876543210，散列表表长为三位，将它分为四组，987|654|321|0，然后将它们叠加求和987 + 654 + 321 + 0 = 1962，再求后3位得到散列地址962。
- 折叠法事先不需要知道关键字的分布，适合关键字位数较多的情况。

处理散列冲突的方法

冲突就是不一样的键却得到了一样的散列值，所以要解决

最常用：开放定址法

下标	0	1	2	3	4	5	6	7	8	9	10	11
关键字	12	25			16			67	56			

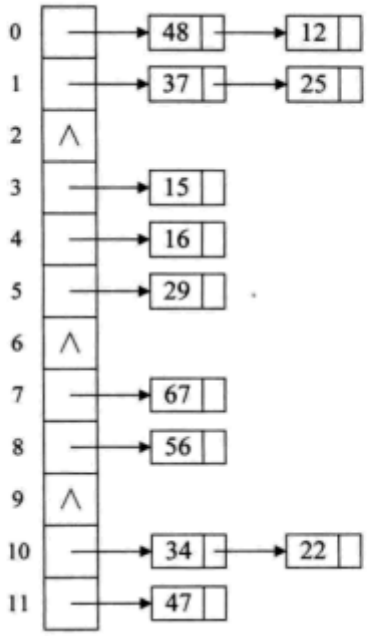
- 当某个键散列值也为1时候，冲突产生，就往后面放，直到找到--线性探测法
- 加入探测位置平方运算，从冲突值两边开始查找---二次探测
- 对偏移量使用随机函数得到---随机探测

再散列法

- 对散列值冲突的，再使用散列函数--包含上面所有随机

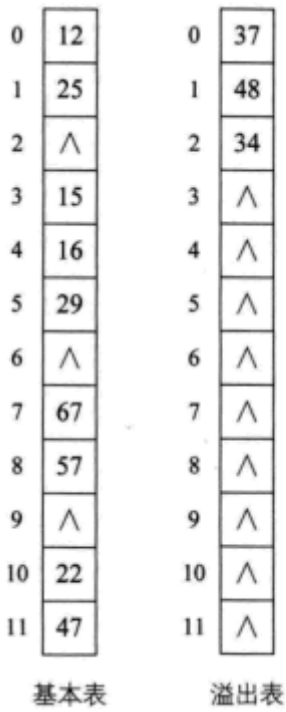
- 这种方法能够使得关键字不产生聚集，但相应地也增加了计算的时间。

链地址法



此时，已经不存在什么冲突换地址的问题，无论有多少个冲突，都只是在当前位置给单链表增加结点的问题。

公共溢出区法



- 使用多一个表储存散列值冲突的地方
- 如：在查键值时候，先看这个键散列值是否与基本表的一致，若不是就使用溢出表的
- 有冲突的数据很少的情况下，公共溢出区的结构对查找性能来说还是非常高的。