## 定义：

哈希表是一种以时间复杂度为常数O(1)为平均时间，来执行插入、删除、查找操作的技术。

## 实现：

哈希表的关键点之一：哈希函数（散列函数）的定义。

需要关键字分布均匀。若哈希函数的计算时间过长，可以从业务逻辑的层面考虑，不使用关键字的所有字符。

关键点之二：解决哈希冲突。

链地址法（分离链接法）：

使用邻接表来解决。

优点：负载因子(load factor)可以较大，空间占用少。

缺点：冲突时链表插入申请地址空间需要额外时间开销。查询时遍历链表需要更多时间。

探测散列表：

1. 线性探测法：发生哈希冲突时，使用线性函数，逐个探测直到找到一个空单元。如：f(i)=i。

若若干个元素形成了聚集，则聚集的情况会越来越严重，形成一些区块，称为一次聚集，导致查询插入删除操作性能急剧降低。

1. 平方探测法：为了解决一次聚集问题，可以选择采用此方法，就是使用二次函数作为探测方法，如f(i)=i²。
2. 双散列：f(i)=i\*hash2(x)。在发生冲突时，在距离hash2(x)、2\*hash2(x)…等位置进行探测。需要保证的是hash2方法不能计算出0值。

对于线性探测，随着哈希表中元素个数增多，表的性能会下降。

对于平方探测，只有表的一半可以用作解决冲突的备选位置，一旦表填充超过一半，甚至当哈希表大小不为素数时，填充未达到一半就不能保证一次找到空的位置。

所以对于探测散列而言，负载因子需小于0.5。并且保证表的大小为素数也很重要。

平方探测解决了一次聚集，但仍可能出现二次聚集。双散列付出计算一个额外散列函数的代价解决了聚集问题。

在实际使用过程中，平方探测散列不用使用第二个散列函数，所以可能更简单高效，尤其是在散列函数计算起来比较耗时的情况下。

## 总结：

1. 当key不为短的串或整数时，需要尤其注意哈希函数的选取。
2. 探测散列法的load factor不应该超过0.5。链地址法的负载因子可以趋近于1。
3. 哈希表是无序的，所以在需要顺序时，可以考虑二叉查找树或其它结构。