线性探测法

在开放定址算法里，线性探测法是散列解决冲突的一种方法，当hash一个关键字时，发现没有冲突，就保存关键字， 如果出现冲突，则就探测冲突地址下一个地址，依次按照线性查找，直到发现有空地址为止，从而解决冲突，

例如 关键字集合{7、8、30、11、18、9、14}，散列函数为：H(key) = (keyx3) MOD 7， 设装填因子（元素个数/散列表长度）为0.7，那么 散列表的长度为 10。

关键字（key）集合存放位置分别为：

7 8 30 11 18 9 14
0 3 6 5 5 6 0

由表格知道，这里的7和14、30和9、11和18出现了位置存放冲突。存放key=7时，散列表长度为10的表中其实没有冲突， 因为7是第一个存在到表中的key，所以一定不会有冲突的，所以7对应散列表的地址0。8、30、11存放的地址分别是3、 6 、5，但是到了key=18时候，发现存放的地址为5，而地址5已经存放了key=11,这时发生了地址冲突。根据线性探测法，算法会探测地址5的下一个地址，即地址6，而此时地址6已经存放了key=30，程序继续探测下一个地址，发现地址7位空，此时把key=18存放到地址7处。以此类推，最后得出的散列表为：

最后集合存放位置

0	1	2	3	4	5	6	7	8	9
7	14		8		11	30	18	9	

成功查找率：（1+1+1++1+3+3+2）/ 7

不成功查找率

计算查找不成功的次数就直接找关键字到第一个地址上关键字为空的距离即可， 但根据哈希函数地址为MOD7，因此初始只可能在0~6的位置。等概率情况下，查找0~6位置查找失败的查找次数为：

地址0，到第一个关键字为空的地址2的距离为3，因此查找不成功的次数为3.

地址1， 到第一个关键为空的地址2的距离为2，因此查找不成功的次数为2.

地址2， 到第一个关键为空的地址2的距离为1，因此查找不成功的次数为1.

地址3，到第一个关键为空的地址4的距离为2，因此查找不成功的次数为2.

地址4，到第一个关键为空的地址4的距离为1，因此查找不成功的次数为1.

地址5，到第一个关键为空的地址2(注意不是地址9，因为初始只可能在0~6之间，因此循环回去)的距离为5，因此查找不成功的次数为5.

地址6，到第一个关键为空的地址2(注意不是地址9，因为初始只可能在0~6之间，因此循环回去)的距离为4，因此查找不成功的次数为4.

不成功查找率：（3+2+1+2+1+5+4）/7