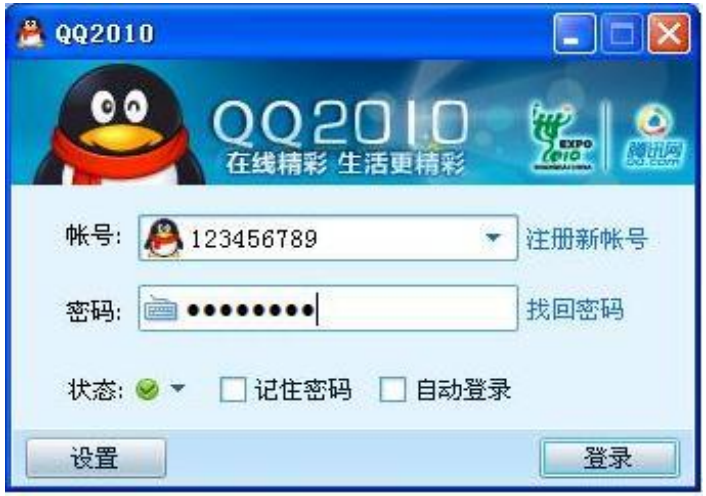
11.1散列表

- 已知的几种查找方法：
  - 顺序查找 $O(N)$
  - 二分查找（静态查找） $O(\log_2 N)$
  - 二叉搜索树 $O(h)$ h为二叉查找树的高度
    平衡二叉树 $O(\log_2 N)$

还有其他方法吗?

[例] 在登录QQ的时候，QQ服务器是如何核对你的身份？面对庞大的用户群,如何快速找到用户信息?



【分析】看看是否可以用二分法查找。

- 十亿（$10^9 \approx 2^{30}$）有效用户，用二分查找30次。
- 十亿（$10^9 \approx 2^{30}$） × 1K ≈ 1024G，1T连续空间。
- 按有效QQ号大小有序存储：在连续存储空间中，插入和删除一个新QQ号码将需要移动大量数据。

【问题】如何快速搜索到需要的关键词？如果关键词不方便比较怎么办？

查找的本质：已知对象找位置。

- 有序安排对象：全序、半序
- 直接“算出”对象位置：散列

❖ 散列查找法的两项基本工作：

- 计算位置：构造散列函数确定关键词存储位置；
- 解决冲突：应用某种策略解决多个关键词位置相同的问题

❖ 时间复杂度几乎是常量：*O（1），即查找时间与问题规模无关！*

❖散列表(哈希表)

类型名称:符号表（SymbolTable）

数据对象集：符号表是“名字(Name)-属性(Attribute)”对的集合。

操作集：Table ∈ SymbolTable，Name ∈ NameType，Attr ∈ AttributeType

1、SymbolTable InitializeTable(int TableSize)：
创建一个长度为TableSize的符号表；

2、Boolean IsIn(SymbolTable Table, NameType Name)：
查找特定的名字Name是否在符号表Table中；

3、AttributeType Find(SymbolTable Table, NameType Name)：
获取Table中指定名字Name对应的属性；

4、SymbolTable Modefy(SymbolTable Table, NameType Name, AttributeType Attr)：
将Table中指定名字Name的属性修改为Attr；

5、SymbolTable Insert(SymbolTable Table, NameType Name, AttributeType Attr)：
向Table中插入一个新名字Name及其属性Attr；

6、SymbolTable Delete(SymbolTable Table, NameType Name)：
从Table中删除一个名字Name及其属性。

- “散列（Hashing）” 的基本思想是：

（1）以关键字key为自变量，通过一个确定的函数 h（散列函数），计算出对应的函数值h(key)，作为数据对象的存储地址。

（2）可能不同的关键字会映射到同一个散列地址上，

即$h(key_i) = h(key_j)$（当$key_i \neq key_j$），称为“冲突(Collision)”。

----需要某种冲突解决策略

[例] 有n = 11个数据对象的集合{18，23，11，20，2，7，27，30，42，15，34}。

符号表的大小用TableSize = 17，选取散列函数h如下：

h(key) = key mod TableSize　(求余)

地址	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
关键词	34	18	2	20			23	7	42		27	11		30		15	

- 存放：

  h(18)=1， h(23)=6， h(11)=11，h(20)=3，h(2)=2, …….

  如果新插入35， h(35)=1, 该位置已有对象！冲突！！

- 查找:

  - key = 22，h(22)= 5，该地址空，不在表中
  - key = 30，h(30)= 13，该地址存放是30，找到！

装填因子（Loading Factor）：设散列表空间大小为m，填入表中元素个数是n，则称α＝ n / m为散列表的装填因子

- α＝11 / 17 ≈ 0.65。

[例] 将acos、define、float、exp、char、atan、ceil、floor、clock、ctime，顺次存入一张散列表中。

散列表设计为一个二维数组Table[26][2]，2列分别代表 2个槽。

如何设计散列函数*h (key) = ?*

h(key) = key[0] – 'a'

acos define float exp char

atan ceil floor clock ctime

	槽 0	槽 1
0	acos	atan
1		
2	char	ceil
3	define	
4	exp	
5	float	floor
6		
……		
25		

如果没有溢出，

$T_{查询} = T_{插入} = T_{删除} = O(1)$