Python算法之旅（第8期）

上期回顾：

描述：小王用计算机生成了10万个1000到1100之间随机整数，请对这些数据进行“去重”和“升序排序”的工作。

函数名：duplicate\_removal\_sort(t):

参数表：t -- 元组，其元素值为1000到1100之间的随机整数。

返回值：一个列表，存储了对元组进行“去重”和“升序排序”后的元素。

示例：对于t=(1095, 1036, 1095, 1036, 1026, 1036)，返回a=[1026, 1036, 1095]

参考代码如下：

def duplicate\_removal\_sort(t):

#把桶清空，因为数据范围是[1000,1100]，故只需101个桶

b = [0] \* 101

for num in t: #把数据装入对应桶中

b[num-1000] += 1

a = []

for i in range(101): #依次把数据从桶里倒出来

if b[i] > 0:

a.append(i+1000)

return a

高考VB算法题源于生活，是对生活实际应用的高度抽象和简化。各种联考模拟卷中也出现了大量源于生活的好题目，今天我们就来欣赏其中一道好题。

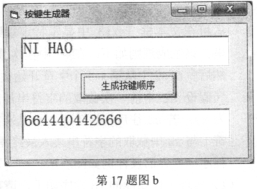
题目：手机数字键盘（2018年6月浙江省名校联盟创新卷1第17题）

难度：3星 有趣：4星 有用：4星

分类：模拟，字符串，字典

描述：老年机因其较大的按键，很适合老年人使用，但其中英文字母的输入方式比较麻烦，导致很多老年人不太会用。如右图是一款老年机的键盘，其字母的输入方式如下：

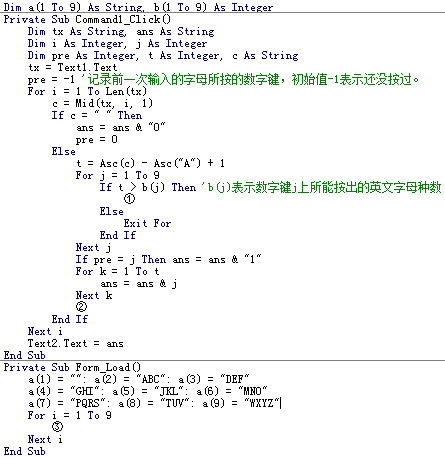
（1）若要输入英文字母“A”,则2键按1下；若要输入“B”，则2键按2下；其他英文字母的输入方式同理；

（2）若连续输入的英文字母在同一数字键中，则在输入下一个英文字母前，需先按下1键以表示确定；若连续输入的英文字母不在同-数字键中，则不需要按1键，直接按所要输入英文字母对应的数字键即可。

（3）若要输入空格，则按0键；连续输入空格，就连续按0键即可。

小陈依据该手机的字母输入规则，设计了如第17图b的VB程序，在Text1中输入要打出的一串字母(只包括大写英文字母和空格)，点击“生成按键顺序”按钮，在Text2中显示需要的按键顺序。请回答下列问题：

1. 若要输入“DI TU" ,则手机数字键的按键顺序为：
2. 请在划线处填上合适的代码



答案：（1）344408188 （1分）

（2）①t = t - b(j) （2分）

②pre = j （1分）

③b(i) = Len(a(i))（2分）

算法分析：

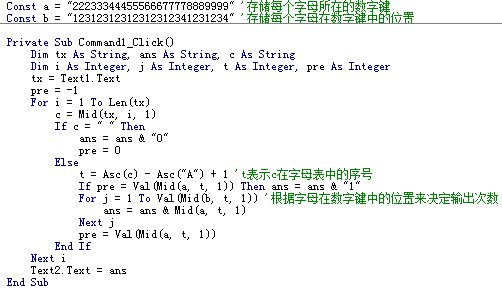
本题是一道源于生活实际的好题，算法也很巧妙。题目设置了字符串数组a来存储键盘上各个数字对应的字母，然后用数组b来存储每个数字对应字母的数量，这是对实际问题的直接模拟；变量t初始化为字符c在字母表中的序号，再通过若干次减法运算，使t指向字符c在键盘中的位置（对应数字j的第t个位置），最后输出结果。

应该说题目提供的算法是很巧妙的，尤其是定位字符c的方法颇具技巧。但就是因为技巧性太强，代码不够浅显易懂，给学生的解题带来较大困难。此外，字符串数组a的出场过于短暂，它的出现似乎仅仅是为了引出数组b，没有发挥它应有的价值。

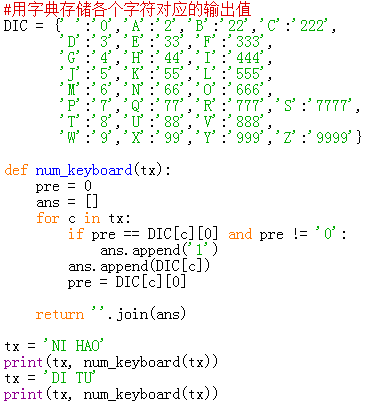
如果不拘泥于对数字键盘的直接模拟，把数据结构做一点简单的改进，我们可以得到一个简单明了的算法。

我们可以直接定义两个长度均为26的字符串常量a和b，分别用来依次存储26个字母所在的数字键和其在数字键中的位置，这样就可以很方便地找到字母的位置了。

代码如下：



曾有人开玩笑地说：“Python企图用字典装载整个世界”，由此可见Python语言中字典对象的应用之广。我们现在就用Python的字典来把它简单粗暴地解决掉：



课后思考：

看了上面的文字，是不是对Python的字典产生了一种刮目相看的感觉？有没有一试身手的渴望？接下来的一道思考题可以满足你的要求。

石头剪子布，是一种猜拳游戏。起源于中国，然后传到日本、朝鲜等地，随着亚欧贸易的不断发展它传到了欧洲，到了近现代逐渐风靡世界。简单明了的规则，使得石头剪子布没有任何规则漏洞可钻，单次玩法比拼运气，多回合玩法比拼心理博弈，使得石头剪子布这个古老的游戏同时用于“意外”与“技术”两种特性，深受世界人民喜爱。

游戏规则：石头打剪刀，布包石头，剪刀剪布。

现在，需要你写一个程序来判断石头剪子布游戏的结果。

函数名：rock\_paper\_scissors(s1, s2):

参数表：s1,s2-- 字符串，分别表示甲乙两个人的出拳，只可能取值在{"石头","剪刀","布"}中。

返回值：字符串，存储了游戏结果，只可能取值在{"甲胜","乙胜","平局"}中。

示例1：s1="石头"，s2="剪刀"，返回"甲胜"；

示例2：s1="石头"，s2="布"，返回"乙胜"；

示例3：s1="石头"，s2="石头"，返回"平局"。

另外，如果你有更 Pythonic（优雅的、地道的、整洁的）代码，或者与本文不同的算法思路和代码实现，请你一定留言或联系我，让我们一起讨论，共同进步。