Python算法之旅（第7期）

上期回顾：

描述：找出字符串中第一个只出现一次的字符

函数名：first\_one(s)

参数表：s -- 被访问的字符串。

返回值：返回第一个只出现一次的字符，如果不存在返回"No"。

示例: s="asdfasdfo"，返回"o"。

参考代码如下：

def first\_one(s):

lib = [0] \* 256

for e in s:

lib[ord(e)] += 1

for e in s:

if lib[ord(e)] == 1:

return e

else:

return "No"

这道题目事实上使用了“桶排序思想”。那么什么是桶排序呢？我们今天就来聊了这个话题。

题目：桶排序思想

难度：2星 有趣：3星 有用：3星

分类：排序

描述：小王想对浙江省30万考生的技术成绩进行降序排序，成绩为0到100之间的整数。由于人数较多，使用选择，冒泡，插入等简单排序算法效率实在太低，是否有更高效的方法？

算法分析：

因为待排序的成绩为0到100之间的整数，所以可以使用桶排序。

桶排序算法分为三个步骤：把桶清空，把数据装入对应桶中和依次把数据从桶里倒出来。

假设我们把成绩存储在列表a中。可以设置一个长度为101的列表b，表示总共有101个桶，它的下标0-100就是各个桶的编号。

第一步：把桶清空，即初始化列表b的值均为0。

第二步：把数据装入对应桶中，即累计列表b各元素的值。例如，若出现a[i] = 80，则b[80] += 1；若最终考80分的学生有10000名，则b[80] = 10000。

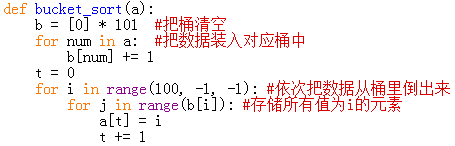
第三步：依次把数据从桶里倒出来，即逆序遍历列表b，从高到低依次输出成绩。若b[i] = k，则表示成绩为i的学生总共有k个，需要将成绩i输出k次。

对应Python代码如下：

函数名：bucket\_sort(a)

参数表：a -- 待排序列表，其元素值为0到100之间的整数。

返回值：该方法没有返回值，但是会对列表的对象进行降序排序。



课后思考：

桶排序被誉为“最简单高效的排序算法”，但是使用它是需要有一定限制条件的，首先是待排序元素必须是非负整数（如果不是，则先将其与非负整数建立一一映射关系），其次是要有足够数量的桶。

如果觉得自己已经理解桶排序思想了，可以试试下面这道题目。

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]

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。