Python练习题

入门级别

1. 【Q01】求用十进制，二进制，八进制表示都是回文数的所有数字中，大于十进制20的最小值。
2. 【Q02】数列的四则运算

   在一串数字中加入适当的运算符，使其结果和是原来的数字的翻转，同时要遵循四则运算的顺序进行（先乘除，后加减。）某些位数之间可以没有运算符，但至少要插入一个运算符。

   例如：在100~999，符合条件的有一下几种情况：

   351 → 3 * 51 = 153

   621 → 6 * 21 = 126

   886 → 8 * 86 = 688

   问题：求位于1000~9999，满足上述条件的数。

   > 解题思路：
   >
   > 要将一个4位数分割开，并在其中添加运算符，然后对该算式进行运算。这里需要要到Python的eval()函数。
   >
   > 同时要考虑到不符合运算规则的算式，如：1+0+01,1 /0 + 11等

   ```
   def Q02():
   ```

```
    oplist = ['+', '-', '*', '//', ''] # 这里用到了//是Python中的整除运算，‘’为没有运算符。
    for i in range(1000,10000): # 从1000开始循环到9999
        num_str = str(i) # 将i转换成字符串，方便后面的运算
        for op1 in oplist: # 第一个运算符
            for op2 in oplist: # 第二个运算符
                for op3 in oplist: # 第三个运算符
                    val = num_str[0] + op1 + num_str[1] + op2 + num_str[2] + op3 +
num_str[3]
                    try:
                        temp = str(eval(val)) # temp为拼接后的字符串的运算结果，需要转换成
字符串，以便后面翻转。
                        if num_str == temp[::-1] and len(val) >4:# 因为题目要求至少有一
个运算符
                            print("符合条件的数字为：{},表达式为{},其运算结果为:
{}".format(num_str,val,temp))
                    except:
                        pass # 对算式不正确的不进行任何处理

if __name__ == "__main__":Q02()
```

这里用的比较暴力的办法，其实分析题目后可以得到，除了用乘法，其他的运算都无法达到目的。

用加法最大的值为：999+9 = 1008，反转时不可能得到原始值的。

3. 【Q03】考拉兹猜想(改版)

   考拉兹猜想：

   对自然数n 循环执行如下操作。 ·n 是偶数时，用n 除以2 ·n 是奇数时，用n 乘以3 后加1 如此循环操作的话，无论初始值是什么数字，最终都会得到1（会进入1 → 4 → 2 → 1 这个循环）

   这里我们稍微修改一下这个猜想内容，即假设初始值为偶数时，也用n乘以3后加1，但只是在第一次这样操作，后面的循环操作不变。而我们要考虑的规则在这个条件下最终又能回到初始值的数。

   例如：2→7→22→11→34→17→52→26→13→40→20→10→5→16→8→4→2

   问题：求在小于10000 的偶数中，像上述的2 这样“能回到初始值的数”有多少个。并打印出来。

```
def is_loop(n):
    '''判断是否符合条件'''
    check = n * 3 + 1 # 本题为改版，第一次都乘以3然后加1
    while check != 1: # 判断是否回到了1
        if check % 2 == 0: # 判断是否为偶数
            check = check // 2 # 这里因为不需要小数，所以用//
        else:
            check = check * 3 + 1

        if check == n: # 判断是否和初始值相等
            return True
    return False # 不符合条件

if __name__ == "__main__":
    num_list = [] # 定义一个用来保存结果的列表
    for i in range(2, 10000, 2): # 因为判断的是偶数，所以从2开始，每次加2
        if is_loop(i): # 符合猜想
            num_list.append(i)
```

```
    print("符合猜想的共有{}个".format(len(num_list)))
    for n in num_list:
        print(n)
```

符合猜想的共有34个。 其实无论把数值调到多大，永远都是这34个，神奇吧。