1 平方和

小明对数位中含有2、0、1、9的数字很感兴趣，在1 到 40 中这样的数包括1、2、9、10至32、39和40，共28个，他们的和是 574，平方和是14362。
注意，平方和是指将每个数分别平方后求和。
请问，在 1 到 2019 中，所有这样的数的平方和是多少？

In [7]:

```
n = 0
s = 0
ss = 0
for i in range(1, 2020):
    if "2" in str(i) or "0" in str(i) or "1" in str(i) or "9" in str(i):
        n += 1
        s += i
        ss += i**2
print(n, s, ss)
```

```
1761 1905111 2658417853
```

In [2]:

```
n = 0
s = 0
ss = 0
for i in range(1, 41):
    if "2" in str(i) or "0" in str(i) or "1" in str(i) or "9" in str(i):
        n += 1
        s += i
        ss += i**2
print(n, s, ss)
```

```
28 574 14362
```

2 字符串移位包含问题

给定两个字符串s1和s2，要求判定s2是否能够被s1做循环移位(rotate)得到的字符串包含。
例如，给定s1=AABCD和s2=CDAA，返回true；给定s1=ABCD和s2=ACBD，返回false。

作业中可以只写优化后的方法与代码，但这种方法不是凭空就能想出来的，前面的思路要看一遍

2.1 枚举

In [13]:

```
s1 = 'ababc'
s2 = 'bca'
n = len(s1)
m = len(s2)
print(m,n)
print(s1,s2)
print(s2 in s1)
for i in range(n):
    if i + m <= n:
        s = s1[i:i+m]
        print(s)
    else:
        s = s1[i:] + s1[:m-(n-i)]
        print(s)

    if s2 == s:
        print(True)
        break

else:
    print(False)
```

```
3 5
ababc bca
False
aba
bab
abc
bca
True
```

In [ ]:

```

```

2.2 生成旋转词，逐一对比

In [5]:

```
s1 = 'AABCD'
s2 = 'CDAAE'
n = len(s1)
for i in range(n):
    s = s1[i:] + s1[:i]
    print(s)
    if s2 in s:
        print(True)
        break
else:
    print(False)
```

```
AABCD
ABCDA
BCDAA
CDAAB
DAABC
False
```

2.3 优化后的思路

对循环移位之后的结果进行分析。
以S1 = ABCD为例，先分析S1进行循环移位之后的结果，如下所示：
ABCD--->BCDA---->CDAB---->DABC---->ABCD
假设我们把前面移走的数据进行保留，会发现有如下的规律：
ABCD--->ABCDA---->ABCDAB---->ABCDABC---->ABCDABCD……
可以看出，对S1做循环移位所得到的字符串都将是字符串S1S1的子字符串。
如果S2可以由S1循环移位得到，那么S2一定在S1+S1上，这样时间复杂度就降低了。

In [6]:

```
s1 = 'AABCD'
s2 = 'CDAA'
if s2 in s1+s1:
    print(True)
else:
    print(False)
```

```
True
```

In [7]:

```
s1 = 'AABCD'
s2 = 'ACBD'
if s2 in s1+s1:
    print(True)
else:
    print(False)
```

```
False
```

3 特殊回文数

【资源限制】时间限制：1.0s 内存限制：512.0MB
【问题描述】

123321是一个非常特殊的数，它从左边读和从右边读是一样的。
输入一个正整数n， 编程求所有这样的五位和六位十进制数，满足各位数字之和等于n 。
【输入格式】输入一行，包含一个正整数n。
【输出格式】按从小到大的顺序输出满足条件的整数，每个整数占一行。
样例输入：52
样例输出：
899998
989989
998899
【数据规模和约定】1<=n<=54

三种思路都要弄明白，在后面两种思路中选择一种你更能独立写出代码的方法。

3.1 暴力解法

遍历所有五位数和六位数

In [3]:

```
# 求各位数字之和判断是否等于n
# 接着将数字转成字符串，再逆转，然后判断是否为回文数
n = int(input())
for i in range(10000,1000000):
    num = str(i)
    result = 0
    for j in num :
        result += int(j)
    if result == n and num == num[::-1] :
        print(num)
```

```
52
899998
989989
998899
```

结果：严重超时！
原因：有两重循环，并且对于不是“特殊回文数”的数，也进行了求和操作，做了很多无效运算。

3.2 先判断是回文数，再求和

In [4]:

```
n = int(input())
for i in range(10000,1000000) :
    num = str(i)
    if num == num[::-1] :
        result = 0
        for j in num :
            result += int(j)
        if result == n :
            print(num)
```

```
52
899998
989989
998899
```

3.3 效率更高的思路

In [6]:

```
n = int(input())
nums = []
for i in range(100,1000) :
    # 针对6位数
    s = str(i) + str(i)[::-1]
    s1 = list(s)
    s2 = [int(x) for x in s1]
    if sum(s2) == n :
        nums.append(int(s))

    # 针对5位数
    s = str(i) + str(i)[:2][::-1]
    s1 = list(s)
    s2 = [int(x) for x in s1]
    if sum(s2) == n :
        nums.append(int(s))

for i in sorted(nums) : # 排序
    print(i)
```

```
52
899998
989989
998899
```

In [ ]:

```

```