Q02 数列的四则运算

IQ：70　目标时间：10分钟

大家小时候可能也玩过“组合车牌号里的 4 个数字最终得到 10”的游戏。

组合的方法是在各个数字之间插入四则运算的运算符组成算式，然后计算算式的结果（某些数位之间可以没有运算符，但最少要插入 1 个运算符）。

例）1234 → 1 + 2 × 3 – 4 = 3
　　9876 → 9 × 87 + 6 = 789

假设这里的条件是，组合算式的计算结果为“将原数字各个数位上的数逆序排列得到的数”，并且算式的运算按照四则运算的顺序进行（先乘除，后加减）。

那么位于 100~999，符合条件的有以下几种情况。

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。

加入运算符倒是不难，难的是如何计算算式吧。

利用编程语言内置的函数或者功能就很简单哦。

思路

解决这个问题时，“计算算式的方法”会影响实现方法。如果要实现的是计算器，那么通常会用到逆波兰表示法[①]，而本题则是使用编程语言内置的功能来实现更为简单。

很多脚本语言都提供了类似 eval 这样的标准函数。譬如用 JavaScript 实现时，可以用代码清单 02.01 解决问题。

代码清单 02.01（q02_01.js）

```
var op = ["+", "-", "*", "/", ""];
for (i = 1000; i < 10000; i++){
  var c = String(i);
  for (j = 0; j < op.length; j++){
    for (k = 0; k < op.length; k++){
      for (l = 0; l < op.length; l++){
        val = c.charAt(3) + op[j] + c.charAt(2) + op[k] +
              c.charAt(1) + op[l] + c.charAt(0);
        if (val.length > 4){ /* 一定要插入 1 个运算符 */
          if (i == eval(val)){
            console.log(val + " = " + i);
          }
        }
      }
    }
  }
}
```

第 10 行中的 eval 就是本题的关键点，接下来只是选择和设置运算符了。虽然有比较深的循环嵌套，但只要确定了位数就没有问题。

的确，如果只是对比和评估字符串的算式，这样实现就足够了。

我发现一旦用了“*”以外的任意运算符，最终的结果就凑不够4位数了。

说得很对。用“+”时，最大的值只有999＋9＝1008。逆序排列不可能得到原始值。当然，用“−”也不可能。

① 逆波兰表示法（Reverse Polish notation，RPN）也称逆波兰记法，是由波兰数学家扬·武卡谢维奇于 1920 年引入的数学表达式，在逆波兰记法中，所有操作符置于操作数的后面，因此也被称为后缀表示法。

基于这样的考虑，如果把代码第 1 行的 op 变量设置成以下值，可以进一步提高程序执行效率。

```
var op = ["*", ""];
```

Point

如果用其他语言实现同样逻辑，需要对 0 进行特别处理。例如在 Ruby 中，“以 0 开头的数”会被当作八进制数来处理，因此必须排除以 0 开头的数。此外，也需要排除除数为 0 的情况。

我打算用C语言来实现一遍，但发现没有eval这样的函数。

很多脚本语言都提供eval这样的函数，但C语言里没有这类功能。这种情况下，可以使用“逆波兰表示法”等实现算式计算。

答案 5931（5 * 9 * 31 = 1395）