D2程序文档

一.第一题：

1.题目说明：

24点游戏，输入程序的数恒为4个，且范围限定在1~13之间的自然数。输入后，将4个数存在一个vector中，保持其顺序不变。

2.运行测试：

由于没有官方的答案，所以我和同学的代码跑了一样的5份数据，发现结果相同。

3.代码分析：

本题目的主要思想是遍历和循环嵌套，即按照运算式的形式生成合法的运算式，并根据不同种类的运算式计算出其结果，结果正确则予以保留，生成代数式并存入set中（这里选择set的原因是set具有天然去重的功能，并且可以自动按照字典序排序。我本来写了一个函数专门去重和排序，经室友@吴海隽提醒直接使用了set）

嵌套主要分以下几层：最外层四层循环用于对a,b,c,d四个输入的数字进行全排列。经过归纳，我发现计算式主要分为五种形式：

```
((A?B)?C)?D型
A?(B?(C?D))型
A?((B?C)?D)型
(A?(B?C))?D型
(A?B)?(C?D)型
```

将他们对应0-4编码，使用一层循环对计算形式进行遍历；最后内部再使用三层循环对三个运算符号进行排列，最后生成24x5x64=7680种可能的计算模式。再通过`GenerateAnswer`函数对结果进行运算和判断。最后遍历set以输出计算式

由于其中可能会遇到中间的除法，为了正确运算我才用了double数1据类型，并且定义了宏EPSILON=0.001,当生成结果和TARGET的差值小于EPSILON时即认为合法。

4.难点疑点：

我认为这道题的主要难点在于归纳总结计算式的模式和不重不漏地遍历所有的可能性。其次就是程序的复用性。本题的代码内，由于采取了归纳模式的想法，对于不是四张牌的情况并不能直接迁移；而对于目标不是24的情况，只需修改宏定义TARGET即可。

二.第二题：

1.题目说明：

24点plus，不固定运算目标和输入个数。

2.运行测试：

ppt中给的两组案例成功通过，和同学的的代码跑了同样的的数据发现生成结果相同。

3.代码分析：

本题的思想依然是遍历。我本来想使用DFS+剪枝的搜索方法，但是考虑到需要计算大于目标的最小值，则一定需要遍历所有的状态空间，而由于只用两种运算符，状态空间数最多为2^23=8388608，因此采取了直接遍历的方法。

在设计遍历时我发现，本题只有两个运算符，因此可以用一串二进制编码来区分不同的状态，（用0代表加法，1代表乘法）例如对于六个数字，二进制序列`100101`和`0101010`则代表了两种不同的状态。由于二进制编码不容易遍历，我设计了一个十进制转二进制的编码函数`ToBinaryCode`，因此只需一个从0到2^1-1的循环就可以遍历状态空间，对应不同的序列生成不同的计算式并计算出结果，和目标值与“当前大于目标值的最小值”对比。最后输出结果。

4.难点疑点：

我认为本题较上一题反而简单一些。只需要想好如何遍历状态空间即可。

此外我认为，如果本题目没有要求输出大于目标值的最小值，可以随机从中间某状态开始计算，并对状态空间进行剪枝，例如从状态000000到111111总体上运算结果在增大，当某状态下计算值已经超过目标值，可以考虑不再向下计算，回头寻找较小的计算结果。这样可以降低成本。

三.感想：

本次作业出了一些问题，我之前写好了代码，但由于拖延症发作一直没有写这个文档。结果在ddl当天晚上骑车时摔了一跤，电脑直接挂掉了，因此没能按时交上作业，这是一个惨痛的教训，告诉我以后不要拖延，早写早交。

另外的一点是这是我第一次注意自己的代码规范，在课后我搜索了大名鼎鼎的Google编程规范，对代码风格注释风格进行了学习和规范，感觉自己的代码可读性得到了很大的提升，希望在以后的代码中能更熟练的完成这一点。

最后我想说的是，因为我之前来自电子系，这次作业的第二题使用了编码的思想，虽然很简单但是感觉还是收到了原来专业的一些课程的启发，感觉非常开心，也有一种他山之石可以攻玉的感觉。