农夫过河问题的深度优先搜索求解

（采用Visual Studio 2017 15.6.4 Windows SDK 10.0.16299.0）

这是第四次上机作业的一部分，总的来说，做得比较中规中矩，完成了要求的工作。

0、程序要完成的内容

用深度优先算法（肯定要使用栈结构）求解农夫过河问题，并输出结果和过程。

1、最终程序的结构是这样的：

用一个字符的最后4位分别表示农夫，白菜，羊和狼的状态，0表示在原岸，1表示在对岸。对于某个确定的状态，找出所有安全并且没有遍历过的后继可能性进行压栈。然后弹栈，相当于进入更深的一层继续寻找。如果弹栈正好是最终的结果就结束并输出找到的走法。为了保存实际通过的过程，使用了另一个栈存储这个路径。在进入更深层时压栈，在回溯时弹栈，从而得到最终结果。

2、一些遇到的问题和解决方法

首先是栈结构的定义。栈至少应包含每一步的数据和已经填充的长度，压栈和弹栈的方法。对于这个具体问题还要考虑一个元素是否存在于栈中，所以还有一个遍历查找的函数，返回一个布尔型返回值，表示是否存在。还因为有路径储存和分类的问题，也不方便弹栈再压栈来完成这个操作，所以还有一个返回下一个弹栈的内容的“窥视”的函数，方便调用。

然后就是查找的算法。原则上只要把压栈和弹栈做好，只要不停压入新的安全的下一步状态，再弹出状态为现有状态就总可以深度优先找到最终的状态。问题就是如何正确记录这个“树”查找的路径。在没有回溯，一直一层层推进时，只有进行记录当前状态就可以，但是对应没有达到最终状态树枝的回溯比较麻烦。

为此，在没有合适后继的状态下，要找到一个合适的路径回溯的终点。由于在没有后继之后，弹栈弹出的只有两种可能，一种是当前节点或某个上级节点的兄弟（此时对应树枝遍历完毕）。这样，为了保证路径记录正确，为每一步添加一个表示遍历层级的unsigned变量。这样只要不断弹出路径最后的节点，再把路径下一个弹出的层级和当前层级比较，若刚好差1，就说明回到了需要回到的分支。

再有就是判定是否安全和是否遍历过的算法。为此增加一个存放所有进栈过的状态的栈（只压栈，不删除），在正常压栈同时也压入这个记录用的栈。如果一个状态在记录中存在，就不再压入栈中了。

3、一些方法的选择和可以改进的地方

在栈的结构上，我使用了顺序栈，而不是链接。实际上链接的方式应该会更加灵活，并且有些算法会更简单。但是因为存储的内容比较简单，使用链接存储结构，指针会大大增加占用的空间。而且因为包括所有安全和不安全的状态，一共只有16种状态的可能，所以不如用顺序存储来得好了。

在我写的程序中，一共使用了三个栈结构进行运算和存储，一个进行运算，一个存储到达过的状态，一个存储路径。可能可以有算法可以用更少的内存完成同样的功能。

4、总结

这个程序总体来说还算比较方便编写，程序的功能很明了，初始状态和末态没有太多变化，甚至不用和用户交互进行输入就可以开始计算和输出。所以整体上只要解决算法问题就可以完成任务了，而不用过多考虑用户输入的诸多可能性。总的来说应该完成得还好。