**斗兽棋开发文档**

**A.斗兽棋设计思路：**

1. **先打印出地图（包括动物地图和地形图）；**
2. **判断用户输入的指令是否合法；**
3. **然后让动物可以移动；**
4. **编写悔棋和撤销悔棋；**
5. **在移动的基础上让动物之间互相伤害；**
6. **在以上基础上加上一些规则（即判断是否可以执行移动）；**
7. **最后作胜负判断。**

**B.斗兽棋设计思路分析：**

1. **先从.txt文档中提取出动物和地形数据，然后再将它们赋值到二维数组中，然后输出**

**地图**

1. **先判断用户输入的指令是否是某些特殊的指令，如“help”、“exit”等，然后再判**

**断**

**一些其他的错误指令，先缩小到两个字符，然后判断第一个是否是1-8，然后第二个**

**是否是wasd；**

1. **在完成动物移动的功能时，首先找到相应指令动物在二维数组中的位置，然后再根据**

**用户输入的指令进行移动动物；**

1. **完成悔棋与撤销悔棋的功能。**

**C. 程序结构：**

**1.定义需要的变量，mapDate地图，animalDate动物,输入的指令等。**

**2.给出提示，打印出棋盘。**

**3.判断指令**

**While（true）**

**If(player) 根据已定义的boolean变量player判断游戏中行动的是左方还是右方**

**If(order.equals(“undo”))做悔棋：先做出判断，如果输入指令前，地图并没有**

**变化，则给出提示，否则，利用记录棋盘的三维数组**

**打印出上一步的棋盘。同时num--，左右方交换。**

**If(order.equals(“restart”))重新开始游戏，对一些变量重新赋值，num，行动的**

**左方or右方，打印出初始地图，将地图初始化。**

**If (order.eqals(“exit”)) 跳出循环同时退出程序，结束游戏。**

**If(reder.equals(“redo”)) 撤销悔棋：同样先做出判断，如果不能再撤销悔棋给 出提示，重新读取指令，否则，撤销悔棋，打印出悔 棋前的地图。同时num++，左右方交换。**

**If(order.equals(“help”) 打印出菜单，包括各种指令介绍。**

**if() 判断错误指令，如果输入的指令不是重新开始，悔棋**

**等特殊指令，并且不符合游戏中动物运动的指令，给**

**出提示，重新读取指令。**

**如果指令合法的话，定义一个字符串placeOfAnimal 调用函数findIt找到正确**

**指令中动物的位置，赋给定义的字符串**

**Switch() 调用moveAnimals()函数，如果输入的指令不满足指令**

**中动物能运动的范围，给出相应的提示，重新读取指**

**令。否则根据指令，调用函数printMap()打印出指令执**

**行后的地图。Num++，然后调用函数**

**theMapOfEveryStep记录执行指令后的棋盘。**

**If() 判断两方是否有输赢，如果其中一方的小窝被对面占**

**领，或者其中一方的动物全部被吃掉，或者都不能再移**

**动，则判为负，即给出提示另一方胜出。**

**一层循环结束**

**定义函数judgeWheatherToMove(char char1, char char 2, int i, int j)**

**因为动物的运动方向有限制，所以判断指令位置的动物能不能做出**

**指令中的运动。**

**定义函数theMapOfEveryStep(int num 1, char[][][] char1, char[][] char2)**

**用一个三维数组记录每次下棋后的地图的状态，用于悔棋和撤销悔 棋。**

**定义函数printMap(int [][] intMap, char[][] charAnimal)**

**用于打印下棋之后的棋盘。**

**定义函数 findIt(String str,char[][ ]charAnimal )**

**根据输入的指令判断出动物的位置。**

**定义函数 winOrNot1(char[][] charwin)**

**判断某一方是否无动物,判断胜负中的一种情况**

**定义函数 allTheAnimalsWhetherToMove(char[][] animalDate, boolean player)**

**判断有没有动物可移动，判断胜负中的第二种**

**定义函数 moveAnimals(String placeOfAnimal, String order, char[][]animalData)**

**判断输入的指令，并根据指令移动动物，其中调用函数**

**judgeWheatherToMove判断正确的移动位置，对于不合法的指令(包**

**括动物死亡(动物死亡是在findIt函数中判断的)，和不能做出指令中**

**的运动)返回特定的值，在前面的switch判断中接收，如果运动可以**

**执行，那么对地图的数组做出相应的改变。**

1. **关键变量**
   1. **mapData char类型的二维数组用来记录地形地图**
   2. **AnimalData char类型的二维数组用来记录动物地图**
   3. **theMapOfEveryStep char类型的三维数组用来记录开局到现在所有的动物地图**
   4. **Player boolean类型的变量用来判断左右方**
   5. **order String类型的变量用来接收用户输入的指令**
   6. **num int类型的变量用来记录当前步数**
   7. **numMax int类型的变量用来记录用户悔棋前经历过的最大的步数**

**D.编程中遇到的问题：**

1. **输出地图时动物地图和地形地图分开输出，不在一起；**
2. **在移动动物时，一开始只想到一个动物一个动物写，后来发现写了太多行，但是不知道怎么合并、精简；**
3. **总是莫名其妙就出bug，然而自己还不知道是什么bug，看代码也很费劲，无从下手；**
4. **在悔棋的时候我一开始只想到根据指令悔棋，所以到了撤销悔棋时就会出现如果上一步动物被吃掉就无法还原被消灭的动物了；**
5. **撤销悔棋时没考虑撤销悔棋的最大步数，结果一开始撤销多了地图就成空白了；**
6. **动物之间互相伤害的代码写得很长，而且逻辑有点混乱，导致运行时bug连连；**
7. **无子可动的判断只知道加上代码，结果导致代码比原来长了将近一倍；**

**E.解决策略：**

1. **询问助教，助教给了一些提示，然后自己 根据提示相出自己的方法成功解决两个地图不在一起的问题；**
2. **询问同学问他们有什么想法，然后自己总结把代码大大精简了；**
3. **向助教寻求帮助，然后助教告诉了我一些常见的错误提示，而且应大家请求做了一期视频，看了之后受益匪浅，学到了找bug的方法；**
4. **后来学到了数组并理解的数组的真正含义，学会用三维数组了，所以想到用三维数组来记录棋盘，但是只能做到悔一定步数的棋，但是这就已经够了；**
5. **在lab课上助教给了一个方法，在定义一个变量来记录撤销悔棋的步数，但是我觉得我的思路不太清晰，后来自己又思考出一种方法，就是把那个变量改成记录落子的最大步数；**
6. **询问同学室友们的意见，把他们总结起来，然后使代码有序编写，最后解决的我代码的最大问题；**
7. **通过分析五子可动的胜利条件，然后想到可以利用前面判断动物是否可以移动，于是循环扫描动物，一旦有动物就给它四个方向的指令判断它是否能移动，这样将原来多出的代码 删掉并且减少了一定的重复。**