1. 棋盘、棋子显示。

棋盘、棋子均使用图片元件显示，其中棋盘当作背景元件；棋子元件命名使用不同的编号进行区分(从左上到右下依次为0~120），采用循环结构在不同的位置生成棋子以匹配棋盘。



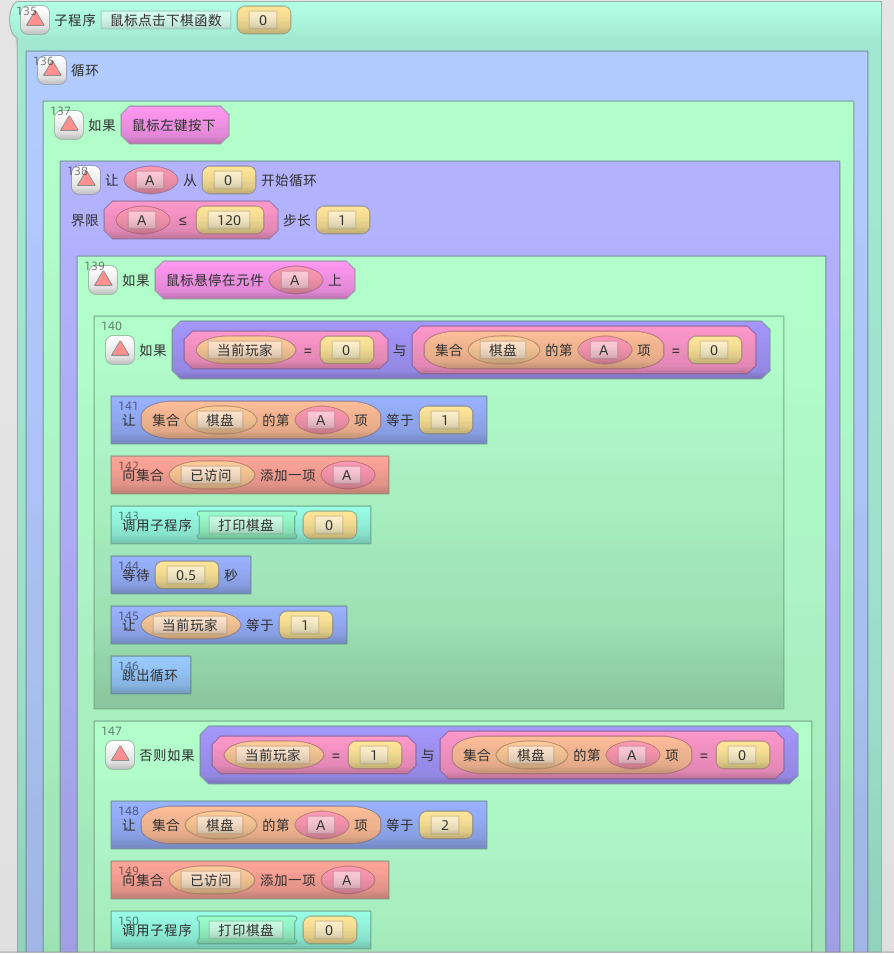
1. 棋盘储存和显示

为了便于棋盘不同棋子颜色的储存，定义一个命名为棋盘的集合储存棋子的颜色（红、蓝或空（1、2或0）），集合的索引为棋子的编号，在每次某一方下完棋后依据该集合进行棋盘的更新。



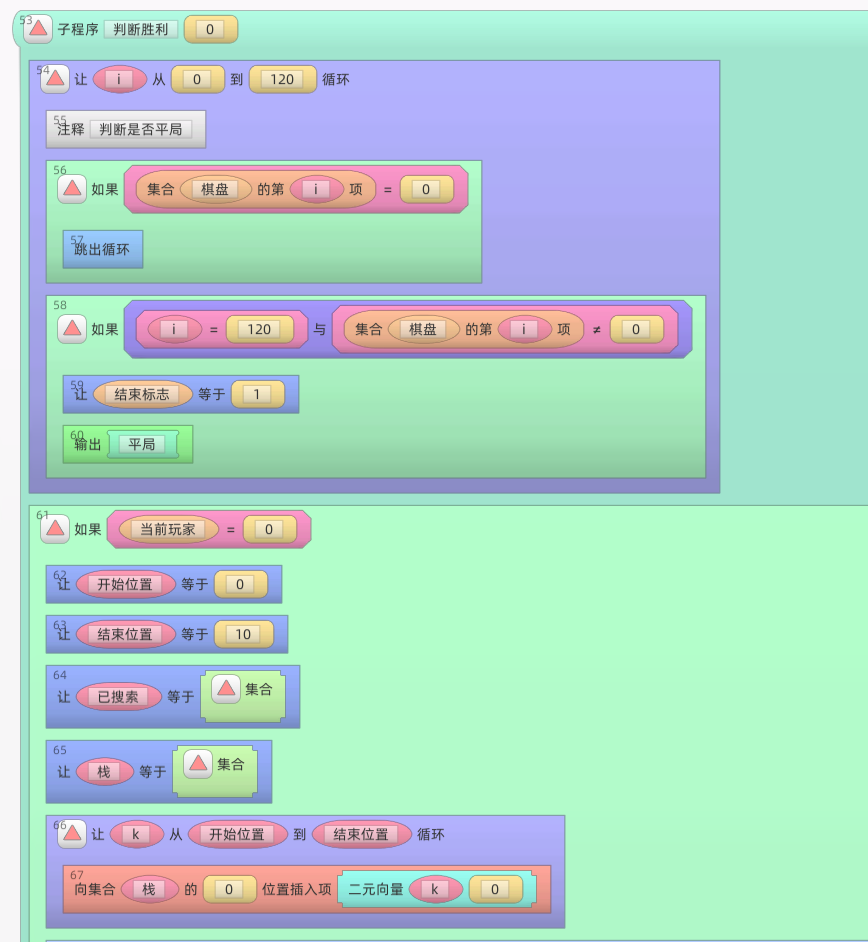
1. 鼠标点击下棋实现

首先定义当前下棋的玩家，用全局变量当前玩家储存，每次有效下棋后进行玩家的替换。当主循环执行到需要玩家下棋时，程序开始循环检测，知道鼠标点击棋盘上的有效位置（当前点击位置棋子为“空”，且点击位置是棋子元件），然后改变棋盘集合进行棋盘显示的更新。



1. 胜利检测

在这个位置采用的栈的思想。首先检查棋盘上是否还有可用位置，如果没有，输出平局。接着，轮流对两位玩家进行胜利检查。对于每个玩家，按照不同的起始位置（蓝色从左侧开始，红色从上侧开始），从起始位置出发，依次遍历所有相邻棋子，标记已访问过的位置。如果找到了棋盘另一侧的棋子，则该玩家获胜。该算法的实现原理是采用深度优先搜索，在每个节点处检查该节点是否为玩家棋子，如果是则标记为已访问过的节点，尝试从该节点往六个方向继续搜索，直到找到棋盘的边缘或者已访问过的节点，然后回溯到上一个节点继续搜索，直到找到一侧的棋子，或者所有节点都被访问过。



1. 主程序循环

定义上一步时刻的棋盘状态，便于添加悔棋功能。

检测当前游戏是否结束，若未结束则输出提示当前下棋的玩家，玩家下玩后进行胜利检测，若有一方胜利，则游戏结束，输出胜利玩家的棋子颜色。

1. 将要开发改善的功能

显示：当前的显示只是输出字符串，以后会做出一个界面，在界面上进行下棋玩家的提示。

美化，功能实现后会对游戏界面进行美化。

人机对战：通过调用外部python程序，进行人机对战的实现。

