总体架构：

一共有棋盘（ChessBoard）、棋子（ChessPiece）、界面（ChessView）、人工智能（AI）四大部分组成。

棋盘类存储着一局象棋游戏所必须的所有信息，包括棋盘上棋子的分布（棋子在棋盘上的坐标）、当前行棋方。并提供象棋游戏的基本操作（行棋、重开、认输\*、求和\*、悔棋\*），在棋局终了时给出胜方，以及为人工智能部分提供当前局面下所有的下一步走法。

棋子类负责管理棋子信息，包括棋子的类型（车马炮士相兵帅）、棋子的阵营（红黑）。并且可以根据当前棋盘的局面，判断走棋是否合法，并可以提供棋子在当前局面下所有可能的走法。

界面类负责程序的控制流跳转，并将棋局的信息展现给用户，监听用户的行为，对用户的行为给出正确的处理。

人工智能类负责人机对弈中机器的决策，包括走棋、是否答应求和\*，是否在适当的时候认输\*。

（\*所标记的功能视情况实现）

各个部分之间的主要通信方式（接口约束）：

0.棋盘与棋子各自的getter和setter方法

棋盘和棋子的所有属性均有getter方法，部分有setter方法，请根据需要自行调用

1.棋盘与界面

一个界面对象中持有一个棋盘对象的引用。

界面通过调用棋盘类中的方法更新棋盘的状态。

**以下涉及的方法均是ChessBoard中的方法**

**开局：public** ChessBoard()

界面类new一个棋盘对象，棋盘对象初始化。

用户行棋：**public** boolean update(Step step)

界面接收到用户点击后，将鼠标点击的坐标转换为象棋棋盘上的坐标（10\*9），交给棋盘的update方法，update方法将返回一个布尔值，代表这一走棋是否合法。返回值若为true，则走棋合法，棋盘将更新这一招走棋，并修改相应的棋局状态，界面则负责将更新后的局面信息展现给用户；返回值若为false，则棋盘不会更新棋局状态，界面也无需做出改变。

AI行棋：**public** Step AIMove()

轮到AI行棋后，界面调用棋盘的AIMove方法，方法返回一个Step对象，是一个合法的走棋棋步，界面根据方法返回的对象更新用户界面。

判断棋局是否终局：**public** State isEnd()

该方法返回一个枚举类型State

**enum** State {***REDWIN***, ***BLACKWIN***, ***DRAW***, ***UNFINISH***}

REDWIN表示此时为红方获胜，BLACKWIN表示此时为黑方获胜，DRAW表示双方打成平手，以上三种情况表示棋局已经终局。UNFINISH表示棋局仍在进行，还没有结果。建议界面在每一次走棋之后（无论是用户还是AI），都调用此方法判断是否终局。

重开：重新new一个棋盘对象即可。

求和\*：**public** **boolean** askForPeace()

界面接收到用户和棋请求后，调用棋盘的askForPeace方法，该方法返回一个布尔值，true为AI同意和棋，false为AI不同意和棋。

悔棋\*：未定

认输\*无需和棋盘进行通信

2.棋盘与AI

棋盘持有一个AI对象的引用。

棋盘通过调用AI类中的方法获得AI的决策。

**以下涉及的方法均是AI中的方法**

让AI走一步棋：**public** Step generateAMove(ChessBoard board)

棋盘调用AI的generateAMove方法，并将棋盘自己传给AI，AI根据当前棋局进行计算，返回一个棋步。

是否答应和棋\*：**public** **boolean** askForPeace(ChessBoard board)

棋盘调用AI的askForPeace方法，并将棋盘自己传给AI，AI根据当前棋局进行计算，判断是否答应对手的和棋请求

是否认输\*：**public** Step generateAMove(ChessBoard board)

（可以通过规定一个特殊棋步，代表AI认输）

AI通过方法传递来的board对象获取棋盘信息。

**以下涉及的方法均是ChessBoard中的方法**

获取下一步所有的合法棋步：**public** List<Step> generateAllNextMove()

3. 棋盘与棋子

棋盘持有若干棋子对象的引用。

棋盘通过调用棋子的方法确定某一特定走棋是否合法，以及产生AI所需要的下一步所有的合法棋步

**以下涉及的方法均是ChessPiece中的方法**

判断走棋是否合法：**public** **boolean** isLegalMove(ChessBoard board, **int** toX, **int** toY)

棋盘调用棋子的方法，由棋子根据自身的类型和当前棋局决定该棋步是否合法，合法返回true，非法返回false

获取棋子下一步所有的合法棋步：

**public** List<Step> getNextMove(ChessBoard board)

棋盘调用棋子的方法，由棋子根据自身的类型和当前棋局找出所有下一步的合法棋步。