五子棋AI：

思路：

遍历棋盘，判断该点是否有子，有棋子则继续遍历。

该点无子时，判断在该点落子后的得分，得分大于800000则落子，否则继续遍历。

若该点的得分大于上一个点的得分，则直接落子在该点。（α-β剪枝）

```
// 下一步棋的位置和对应下法的得分
class PointAndValue {
   public final Point point;
   public final int value;
   public PointAndValue(Point point, int value) {
      this.point = point;
      this.value = value;
   }
}
//获取最大分数的下发
private PointAndValue getMaxEvaluate(int leftStep, int color, int passValue) {//
这个passValue就是其他节点传过来的值
    Point maxPoint = null;
    int maxValue = -Integer.MAX_VALUE;
    for (int i = 0; i < Chessboard.width; i++) {
        for (int j = 0; j < Chessboard.height; j++) {
            Point p = new Point(i, j);
            if (get(p) != 0) continue; // 如果这个点已经有子了，则跳过
            set(p, color);//下棋
            int val = evaluateBoard();//判断一方棋子的得分
            if (val > 800000) {
                set(p, 0);
                return new PointAndValue(p, val);
            }
            if (leftStep > 1) {
               PointAndValue nextStep = getMaxEvaluate(leftStep - 1, 3 - color, -
maxValue);
               if (nextStep.value >= passValue) { // α-β剪枝
                   set(p, 0);
                   return new PointAndValue(p, nextStep.value);
               }
               val = -nextStep.value;
            }
            if (maxPoint == null || val > maxValue) {
               maxPoint = p;
               maxValue = val;
            }
            set(p, 0);
        }
    }
    return new PointAndValue(maxPoint, maxValue);
}
```

```
private int evaluateBoard(int color) {
      int values = 0;
      for (int i = 0; i < Constant.MAX_LEN; i++) {
         for (int j = 0; j < Constant.MAX_LEN; j++) {
            Point p = new Point(i, j);
            // 如果同一个方向有连续5个子，则 values += 1000000;
            // 如果同一个方向有连续4个子并且两边都没有堵住，则 values += 300000;
            // 如果同一个方向有连续4个子并且仅有一边被堵住，则 values += 2500;
            // 如果是“活三”的情况，则values  += 3000
            // 如果是“活三”被堵住了一边，则values += 500;
         }
      }
      return values;
   }
```