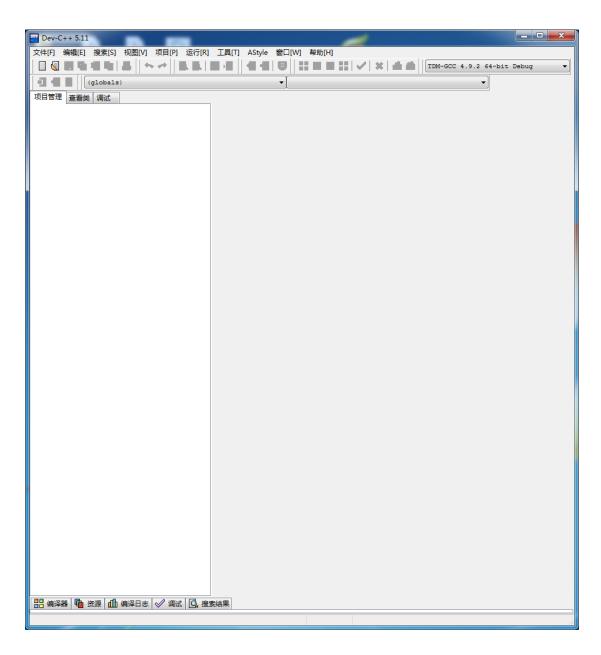
# 开发文档

# 1. 开发环境

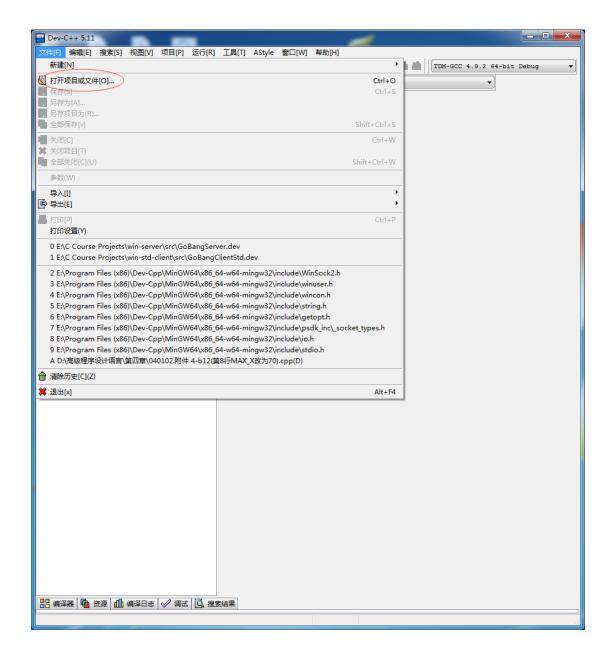
- Windows 7 SP1
- DEV C++ Version 5.11 (<a href="https://sourceforge.net/projects/orwelldevcpp/">https://sourceforge.net/projects/orwelldevcpp/</a>)

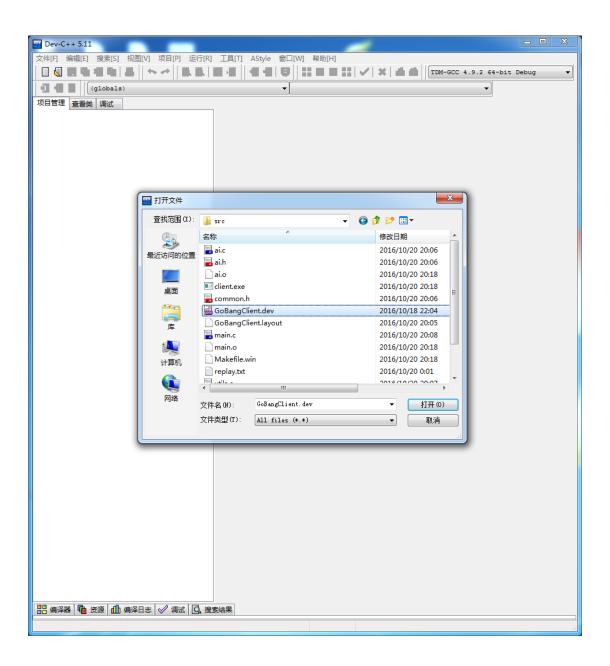
## 2. 配置依赖

## 2.1 启动 DEV C++



#### 2.2 打开 win-server 项目





#### 2.3 打开项目属性

```
_ D X
GoBangServer - [GoBangServer.dev] - Dev-C++ 5.11
                                       [P] 运行[R] 工具[T] AStyle 窗口[W] 帮助[H]
 文件[F] 编辑[E] 搜索[S] 视图[V] 项E
  □ 🔞 📕 🐿 🛍 🛍 🖳 া 新建单元[N]
                                                              (globals)
 项目管理 查看类 调试
                                  ⑤ 项目属性[O] Ctrl+H

☐ GoBangServer

☐ GoBangServer
                                                                  // Judge col

count = 1;

j = y - 1;

while (j >= 0 && board[x][y] == board[x][j])
                                                       88
89
90
      main.c
utils.c
utils.h
                                                       91 = 92 93 94 - 95 96 97 98 = 99
                                                                       --j;
                                                                  j = y + 1;
while (j < BOARD_SIZE \&\& board[x][y] == board[x][j])
                                                      100
                                                                      ++j;
                                                      101
102
                                                                  if (count >= WIN FLAG) return TRUE;
                                                      103
                                                      104
                                                                  // Judge left oblique
count = 1;
                                                      106
                                                                  i = x - 1;

j = y - 1;

while (i >= 0 && j >= 0 && board[x][y] == board[i][j])
                                                      107
                                                      109
                                                     110 = 111 112 113 114 - 115 116
                                                                      --i;
--j;
                                                                  i = x + 1;
                                                      117
118
119 =
                                                                  j = y + 1;
while (i < BOARD_SIZE && j < BOARD_SIZE && board[x][y] == board[i][j])
                                                      120
121
                                                      122
123
124
125
126
127
128
                                                                      ++j;
                                                                  if (count >= WIN_FLAG) return TRUE;
                                                                  // Judge right oblique

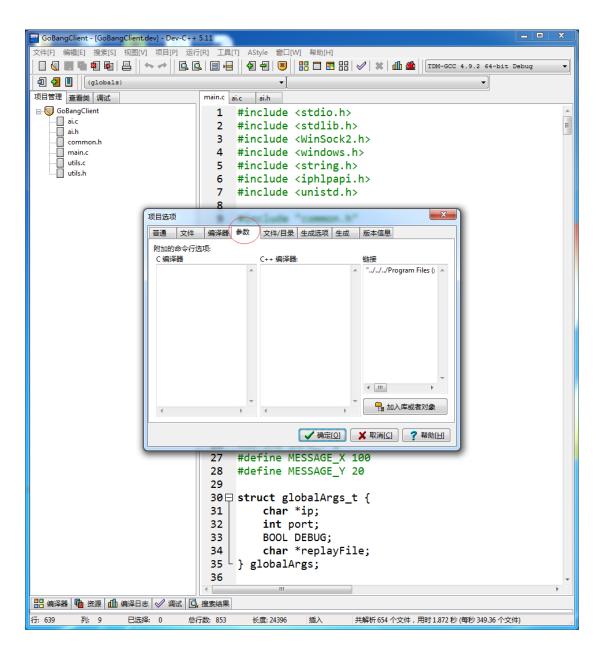
count = 1;

i = x - 1;

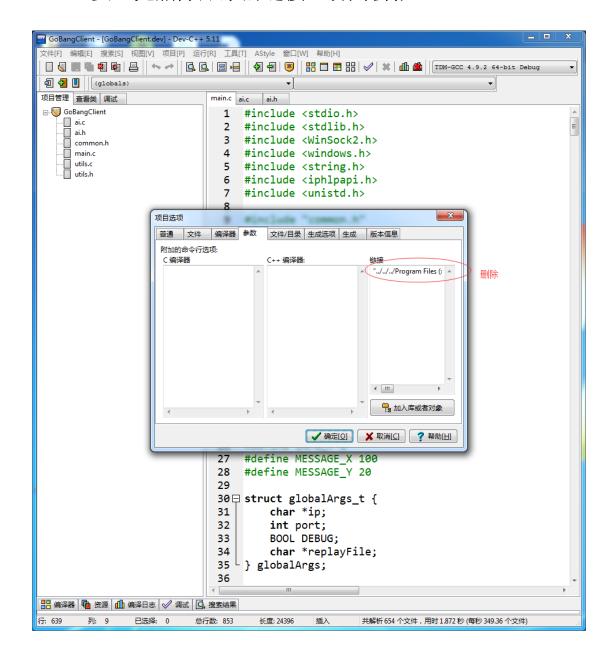
j = y + 1;

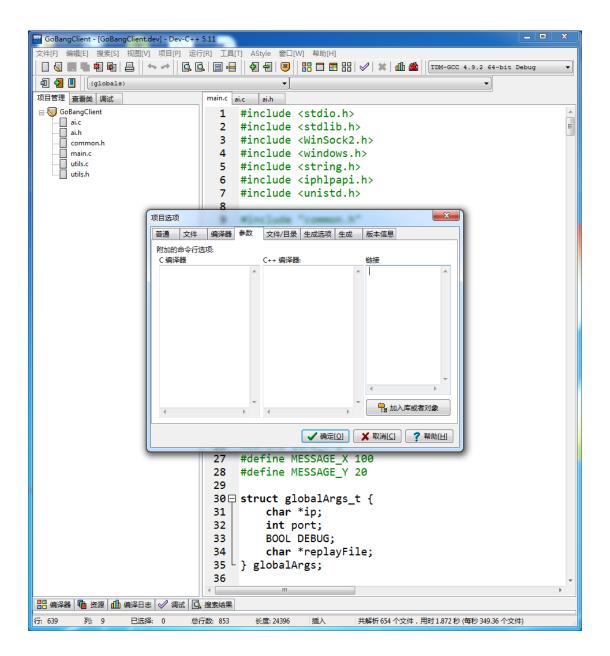
while (i >= 0 && j < BOARD_SIZE && board[x][y] == board[i][j])
                                                     129
130
131
132 =
133
134
                                                                      ++count;
                                                                       --i;
++j;
                                                      135
136
137
                                                                   \begin{array}{l} i = x + 1; \\ j = y - 1; \\ \text{while } (i < BOARD\_SIZE \&\& j >= 0 \&\& board[x][y] == board[i][j]) \\ , \end{array} 
                                                      138
                                                      139
140
                                                      141 <del>|</del>
                                                                      ++count;
行: 64 列: 25 已选择: 0 总行数: 365
                                                                长度: 7577 插入
                                                                                            共解析 602 个文件,用时 1.84 秒 (毎秒 327.17 个文件)
```

#### 2.4 选择"参数"

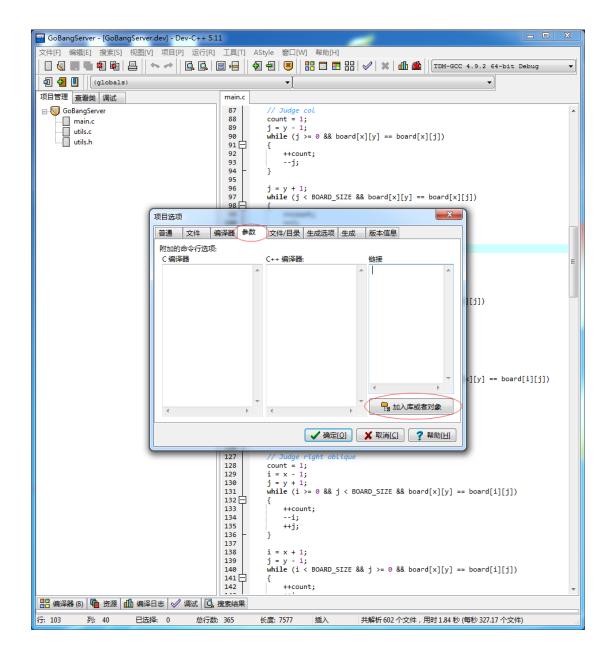


#### 2.5 重要: 先删除原来的链接, 否则会报 file not found error

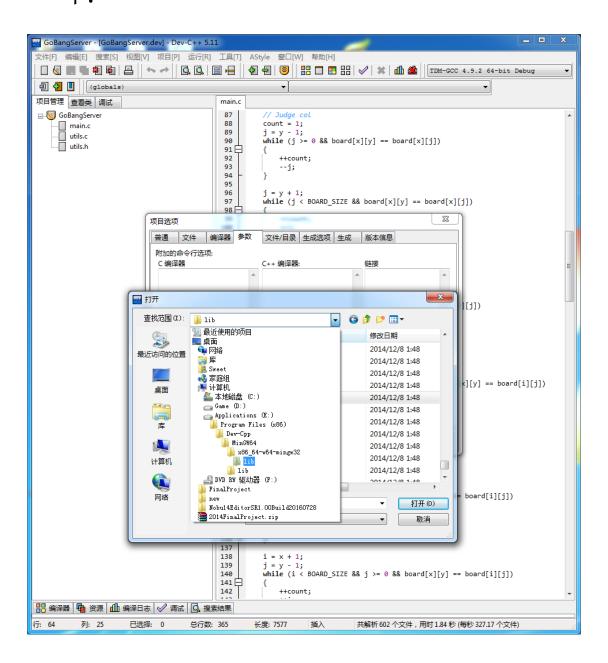




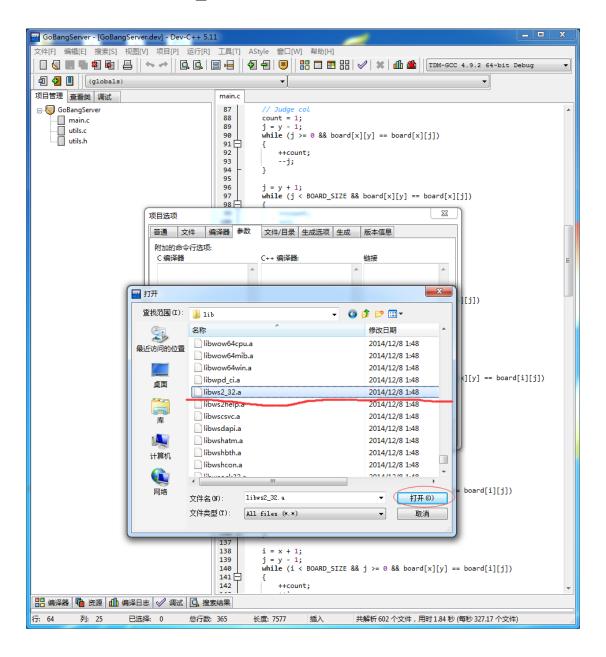
#### 2.6 点击"加入库或者对象"

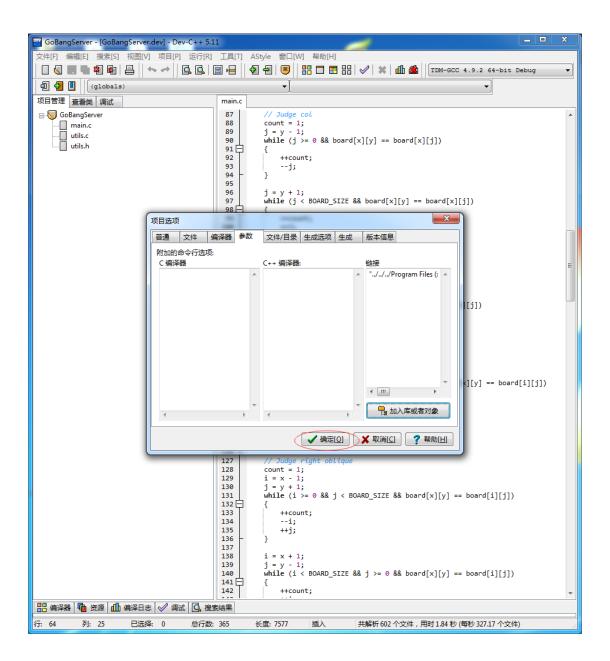


# 2.7 找到安装的 DEV C++目录下的 lib 文件夹,目录树结构如下:



## 2.8 选择 "libws2\_32.a" 文件, 打开, 然后确定





#### 2.9 编译

```
☑ GoBangServer - [GoBangServer.dev] - Dev-C++ 5.11
文件[F] 编辑[E] 搜索[S] 视图[V] 项目[P] 运行[R] 工具[T] AStyle 窗口[W] 帮助[H]
                                                                                                                                      _ D X
  □ 🖁 📕 🐿 🛍 🖳 🖴 → | [4 🔡 编译[C]
                                                                              F10
                                           ■ 运行[R]
 (globals)
                                           編译运行[o]
                                                                        F11
 项目管理 查看类 调试
                                          F12

☐ ■ GoBangServer

                                           ☑ 语法检查[S]
     main.c
utils.c

    检查当前文件语法[S](V) Ctrl+F9

                                                                               && board[x][y] == board[x][j])
                                              参数(W)...
                                              编辑Make文件
                                           一 清除[i]
                                          ■ 性能分析[f]
                                                                              ARD_SIZE \&\& board[x][y] == board[x][j])
                                           🏙 删除性能信息[Z]
                                              切换断点[T]
                                                                         F4
                                           ✓ 调试[D]
                                                                         E5
                                                                         F6 WIN_FLAG) return TRUE;
                                           ¥ 停止执行(Y
                                                                // Judge left oblique count = 1;
                                                   105
                                                   106
                                                               i = x - 1;

j = y - 1;

while (i >= 0 && j >= 0 && board[x][y] == board[i][j])
                                                   107
                                                   109
                                                   109
110 =
111
112
113
114
-
115
116
                                                                    ++count;
                                                                   --i;
--j;
                                                               i = x + 1;
                                                   117
118
119 =
                                                               j = y + 1;
while (i < BOARD_SIZE && j < BOARD_SIZE && board[x][y] == board[i][j])
                                                   120
121
                                                   122
123
124
125
126
127
128
                                                                   ++j;
                                                               if (count >= WIN_FLAG) return TRUE;
                                                               // Judge right oblique

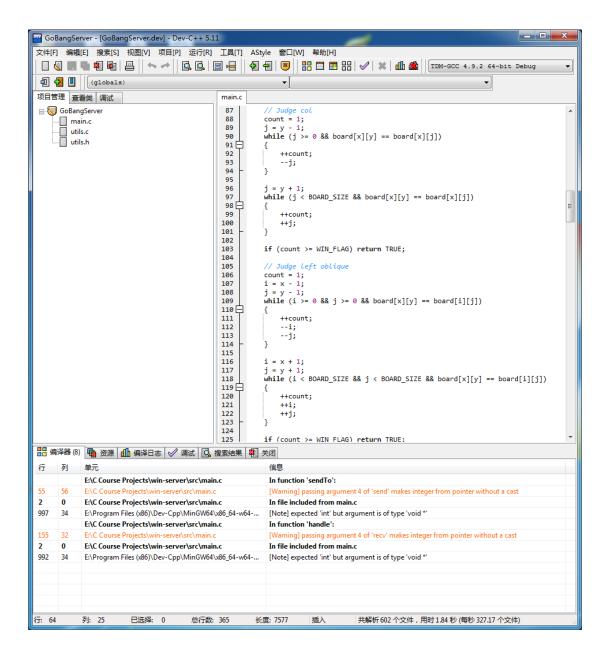
count = 1;

i = x - 1;

j = y + 1;

while (i >= 0 && j < BOARD_SIZE && board[x][y] == board[i][j])
                                                   129
130
131
132 =
133
134
                                                                   ++count;
                                                   135
                                                                    ++1;
                                                   136
137
                                                                \begin{array}{l} i = x + 1; \\ j = y - 1; \\ \text{while (i < BOARD\_SIZE \&\& } j >= 0 &\& board[x][y] == board[i][j]) \\ , \end{array} 
                                                   138
                                                   139
140
                                                   141 <del>|</del>
142 |
                                                                   ++count;
□□ 編译器 □ 资源 □□ 编译日志 | ✓ 调试 □□ 搜索结果
行: 64 列: 25 已选择: 0 总行数: 365
                                                             长度: 7577 插入
                                                                                        共解析 602 个文件, 用时 1.84 秒 (毎秒 327.17 个文件)
```

#### 2.10 没有红色的 error 即表示编译成功



#### 3. 编写 ai

你所需要编写的大部分文件都在 ai.c 文件里:

```
GoBangClient - [GoBangClient.dev] - Dev-C++ 5.11
 文件[F] 编辑[E] 搜索[S] 视图[V] 项目[P] 运行[R] 工具[T] AStyle 窗口[W] 帮助[H]
  (globals)
 项目管理 查看类 调试
                                                        main.c ai.c ai.h
 ■ GoBangClient
                                                        1 #include "ai.h"
       ai.c
ai.h
common.h
main.c
utils.c
                                                            * YOUR CODE BEGIN
* 你的代码开始
*/
                                                            /*
* You can define your own struct and variable here
* 你可以在这里定义你自己的结构体和变量
*/
                                                        10
                                                        11
12
                                                        13
14
15
                                                            * You should init your AI here
* 在这里初始化你的AI
*/
                                                        16
                                                        17
                                                        18 void initAI()
19 ☐ {
                                                        20 T }
                                                        23
                                                              /*
* Game Start, you will put the first chess.
* Warning: This method will only be called when after the initialize ofi the map,
* Or this method will not be called.
* You should return a valid Position variable.
* 撲局开始,首先由你未落子
* 请注意: 只有在当撲局初始化后,轮到你落子时才会触发这个函数,如果在棋局初始化完毕
* 在这里,me的取值只可能是me(1),即board(棋盘)上为me(1)的位置表示你的棋子,board(表
* 你需要返回一个结构如Position,在X属性和X属性填上你想要落子的位置。
                                                        24
25
                                                        26
                                                        27
28
                                                        29
                                                        32
                                                        33 struct Position aiBegin(const char board[BOARD_SIZE][BOARD_SIZE], int me)
34 □ {
35 | /*
                                                                     /*
* TODO: Write your own ai here!
* Here is a simple AI which just put chess at empty position!
* 代徴. 在这里写下你的I。
* 这里有一个示例AI,它只会寻找第一个可下的位置进行落子。
*/
                                                        36
37
38
                                                        39
                                                       40
41
42
43
44
45
46 —
47
48 —
49
50
51
                                                                    int i, j;
struct Position preferedPos;
                                                                         for (j = 0; j < BOARD_SIZE; j++)
                                                                         if (EMPTY == board[i][j])
                                                                             {
    preferedPos.x = i;
    preferedPos.y = j;
    return preferedPos;
                                                        52
53
54
55
                                                                    return preferedPos;
编译器 👊 资源 📶 编译日志 🤣 调试 🗓 搜索结果
            列: 19
                                                                      长度: 2746
                            已选择: 0
行: 15
                                                 总行数: 95
                                                                                                        共解析 654 个文件,用时 1.872 秒 (每秒 349.36 个文件)
```

```
GoBangClient - [GoBangClient.dev] - Dev-C++ 5.11
  文件[F] 编辑[E] 搜索[S] 视图[V] 项目[P] 运行[R] 工具[T] AStyle 窗口[W] 帮助[H]
     (globals)
   项目管理 查看类 调试
                                                                                                                             main.c ai.c
                                                                                                                                                                   ai.h
                                                                                                                             41
42
43
    ■ GoBangClient
                                                                                                                                                          int i, j;
struct Position preferedPos;
                 ai.c
ai.h
common.h
main.c
utils.c
                                                                                                                             43
44
45
46 =
47
48 =
49
                                                                                                                                                          for (i = 0; i < BOARD_SIZE; i++)
    for (j = 0; j < BOARD_SIZE; j++)</pre>
                                                                                                                                                                     {
   if (EMPTY == board[i][j])
                                                                                                                             50
51
52
                                                                                                                                                                                             preferedPos.y = j;
return preferedPos;
                                                                                                                             53
54
55
56
57
58
59
                                                                                                                                                       return preferedPos;
                                                                                                                                           * Game ongoing, the competitor put the chess at the position (otherX, otherY). You

* You should return a valid Position variable.

* 模局进行中、对方上一步落子在(otherX, otherY)的位置。発到你落子了。

* 在这里,哪的取值只可能是NE(1),即board(模盐)上为NE(1)的位置表示你的棋子,board(表

* 你需要返回一个结构如Position,在X属性和X属性填上你想要落子的位置。
                                                                                                                              63
                                                                                                                             | TODO: Write your own ai here!

**Todo: Write your own ai here!

**Here is a simple AI which just out share at a simple AI which it is a simple AI which just out share at a simple AI which it is a simple AI which it is a simple AI which it is 
                                                                                                                                                           /*
* TODO: Write your own ai here!
* Here is a simple AI which just put chess at empty position!
* 代徴. 在这里写下你的AI。
* 这里有一个示例AI,它只会寻找第一个可下的位置进行落子。
*/
                                                                                                                             70
71
72
                                                                                                                             72
73
74
75
76
77 \square
78
79 \square
                                                                                                                                                          int i, j;
struct Position preferedPos;
                                                                                                                                                           for (i = 0; i < BOARD_SIZE; i++)</pre>
                                                                                                                                                                      for (j = 0; j < BOARD_SIZE; j++)</pre>
                                                                                                                              80 T
                                                                                                                                                                                if (EMPTY == board[i][j])
                                                                                                                               82
                                                                                                                                                                                            preferedPos.x = i;
                                                                                                                             83
84
85
86
87
                                                                                                                                                                                             preferedPos.y = j;
return preferedPos;
                                                                                                                              88
                                                                                                                             89
90
91
                                                                                                                                                          return preferedPos;
                                                                                                                             92
93
94
                                                                                                                                           * YOUR CODE END
* 你的代码结束
*/
 列: 37
                                                                已选择: 0
行: 36
                                                                                                               总行数: 95
                                                                                                                                                                长度: 2746
                                                                                                                                                                                                          插入
                                                                                                                                                                                                                                           共解析 654 个文件,用时 1.872 秒 (每秒 349.36 个文件)
```

请参考里面的注释进行代码的编写