// 尝试在第row行放置皇后  
void placeQueen(int row, int &count, int \*queens) {  
 if (row == 16) { // 如果已经放完了所有皇后，则找到了一组解  
 count++; // 统计解的个数  
 return;  
 }  
  
 for (int col = 0; col < 16; col++) { // 尝试在当前行的每一列放置皇后  
 if (isValid(row, col)) { // 如果可以在当前位置放置皇后  
 queens[row] = col; // 在当前位置放置皇后  
placeQueen(row + 1); // 递归到下一行  
queens[row] = 0; // 回溯，撤销当前放置的皇后  
}  
}  
}  
  
// 检查在第row行，第col列放置皇后是否合法  
bool isValid(int row, int col, int \*queens) {  
for (int i = 0; i < row; i++) { // 检查同一列是否有皇后  
if (queens[i] == col) return false;  
}  
for (int i = row - 1, j = col - 1; i >= 0 && j >= 0; i--, j--) { // 检查左上方是否有皇后  
if (queens[i] == j) return false;  
}  
for (int i = row - 1, j = col + 1; i >= 0 && j < 16; i--, j++) { // 检查右上方是否有皇后  
if (queens[i] == j) return false;  
}  
return true; // 如果没有冲突，则合法  
}  
  
int main() {  
 int count = 0;  
 int queens[16] = {0};  
placeQueen(0, count, queens); // 从第0行开始放置皇后  
cout << "共有" << count << "组解" << endl;  
return 0;  
}

算法伪代码为: