**Name: Sheng Liang, Student ID: 301541117**

To solve fifteen puzzle, I divided the solution into three steps: Let’s assume the board size is 8\*8.

|  |  |
| --- | --- |
| 1 | 2 |
| 3 | 4 |
| 5 | 6 |
| 7 | 8 |
| 9 | 10 |
| 11 | 12 |
| 13 | 14 |

2

6

6

2

I write a class Point to store the position of a number in the board.

I used a 2D array to store the board.

I solve this puzzle piece by piece(i.e. first move 1 to its right position in the board, then 2, then 3…)

1)If the number’s right position in the board lies in the yellow region:

First I search the number and located its current position on the board, and then find the route that move the number to its correct position. First I calculate the difference of the two positions’ row and column indexes, then the number should first move horizontally(left or right) then vertically(up), in this case the numbers that have already placed in the right positions won’t be affected. And then I add every step of the route into a LinkedList(let’s say temp). Second, I search and locate the blank space and then move it to the nearby position of the number I am trying to move, and save every step into another LinkedList(let’s say record) . After the blank space is next to the number, then I take out the first element in temp to be position that the blank space to reach. Since the blank space is next to the number, I can rotate the blank space around the number clockwise or counter-clockwise(i.e. if clockwise rotation will affect the numbers that’s in its correct position, then I used counter-clockwise.) After the blank space is in the position that the moving number should move to, then I swap the number and blank space. Repeat all the step until the number is in its correct position. And then move the next number…

2)If the number’s correct position lies in the last two columns but not in the last two row(cyan area):

Similar to case 1, but we move the two numbers in the last two columns together(i.e. 3 and 4, in the board size 4\*4). The reason why I move them together is that if I move 3 in its right position, then it’s impossible to place the blank space in the corner, which should be the correct position of 4, so that it’s impossible to move 4 into its correct position. So I first move 3 into 4’s correct position, and move 4 to the correct position of 8(which mean 4 is under 3 in the last column). then take them as a whole, and move them to their final position. There are two special cases after moving 3 to 4’s final position:

1. 4 has already lies in 3’s final position.
2. 4 lies in 7’s final position and blank space is in 3’s final position.

This time I use a particular moves to swap the positions of the two numbers so that it can be changed to the normal case. (for special case 1): L,U,R,D,L,D,R,U,U,L,D,R,D (for special case 2): R,D,L,D,R,U,U,L,D,R,D

3)If the number’s correct position lies in the last two row but not in the last two row(gray area):

Similar to case 2, I move the two numbers in the same column together to their right positions(i.e. 9,13 in the board 4\*4). So I first search and locate them and move it to the same column. After moving into a same column, if they are in the last two column, then move them to the column on the left. Next, view them as if they are horizontally placed like case 2. And then, they can be moved to their right positions as a whole, or if it’s the special case mentioned above, swap them using the previously mentioned particular moves.

4)Finally for the last 2\*2 board(green area), just keep rotating the four numbers clockwise or counter-clockwise until they are all in the correct positions.

Every move of the blank space is in the LinkedList record, so if the next step of the blank space is moving one direction, then the number in that direction just move oppositely. So that I can export every move of the number from the LinkedList record in to a solution.txt.

The pictures below shows the basic process of the solutions:

图片包含 日程表

描述已自动生成图示, 示意图

描述已自动生成地图

中度可信度描述已自动生成图示

描述已自动生成