Design

Python structure and flow

1.At the start of the game, display a brief introduction about the game.

2.Prompt up the user to enter four letters to represent the four directions. Check whether the input letters are valid (different from each other & is digit). If the input letters are invalid, inform the player to input again.

2.Prompt the user to select either 8 or 15's puzzle. Set two (both 8 and 15's) standard (from 1 to 8 or from 1 to 15) puzzles and randomize them into the initial puzzle given to the user

3.Repeatedly prompt the player to enter sliding direction to move the adjacent tile.

4.After each move, show the updated puzzle on the screen and prompt further direction if needed

5.Inform the user when the puzzle is solved. Prompt the user to continue another game or end the program.

6.Track total number of moves made for each game and have it displayed as the puzzle is solved.

The breakdown of my logic (the problems I meet with and my solution)

1.How to generate a solvable puzzle → use inversion number (This will be explained in detail later)

2.When prompting user to input letters to represent directions, how to inform player of the false input → use while loop to keep the user entering again until the users inputs correct form

3.In the sliding process, how to avoid the situation where the user inputs directions that are not allowed but the adjacent tile will still move → use if statement inside the moving function and inform the user of wrong move. The puzzle will change until he enters right direction that is allowed.

4.How to check whether the puzzle is solvable (at first I used a very complicated method and it didn't work) → just compare the puzzle number with elements inside a standard list.

Different python objects I use to develop my program

1.puzzle_list_nine, puzzle_list_sixteen: global variables
Used to save the number elements of the puzzle.

2.inversion_number_nine, inversion_number_sixteen: local variables
Used to check whether a randomized puzzle is solvable

3.left, right, up, down: local variables
Used to save the input letters as moving directions separately

4.direction_list: local variables
Used to save the input direction letters and check whether they are totally different from each other

5.p_direction: global variables
Used to save the input letters as moving directions as a whole

6.p_nine_direction, p_sixteen_direction: local variables

Used to save user's input direction when playing 8 or 15's puzzle. Use it to instruct further movements of the puzzle.

<u>The logic used to generate the randomized puzzle, both 8 and 15's</u>

1.Main idea:

(1) Set a standard puzzle
(2) Use random.shuffle() to randomize it
(3) Use the method of "inversion number" to check whether the randomized puzzle is solvable

*** The definition of "Inversion number":**

In a list, if the previous number is larger than the latter number (they not necessarily be neighbor), then we call it an "Inversion". The total number of "inversions" in this list is called the "Inversion Number".

2.The 8's puzzle in detail:

(1) First set a standard 8's puzzle ordered from 1 to 8
(2) Use random.shuffle() to randomize it
(3) Use the method of "inversion number":
    (i) There are odd (three) columns in the puzzle.
    (ii) Use two for loop and an if statement to calculate the total number of inversion numbers.
    (iii) If the inversion number can be divided by 2, then the puzzle is solvable.
(4) Print out the puzzle if it is solvable. Otherwise, continue randomizing it until it is solvable.

3. The 15's puzzle in detail:

(1) First set a standard 15's puzzle ordered from 1 to 15
(2) Use random.shuffle() to randomize it
(3) Use the method of "inversion number":
    (i) There are even (four) columns in the puzzle.
    (ii) Use two for loop and an if statement to calculate the total number of inversion numbers.
    (iii) If the inversion number can be divided by 2, and the final space location is even (here:2) lines away from the initial space location: then the puzzle is solvable
    (iv) If the inversion number can't be divided by 2, and the final space location is odd (here:1,3) lines away from the initial space location: then the puzzle is solvable
(4) Print out the puzzle if it is solvable. Otherwise, continue randomizing it until it is solvable.

Functions

1.puzzle_solvable_nine():

Parameter: no parameter
Usage: show the solvable randomized 8's puzzle
(1) Use inversion number to know whether the puzzle is solvable
(2) Use while True to keep looping until we get a solvable puzzle
(3) When we get the correct answer, print out the randomized and solvable puzzle.
(4) Display it as a 3*3 table.

2. **puzzle_solvable_sixteen():**

Parameter: no parameter
Usage: show the solvable randomized 15's puzzle
(1) Use inversion number to know whether the puzzle is solvable
(2) Use while True to keep looping until we get a solvable puzzle
(3) When we get the correct answer, print out the randomized and solvable puzzle.
(4) Display it as a 4*4 table.

3.check_choose_direction(p_direction):
Parameter: p_direction: the input letters representing moving directions
Usage:
(1) Check whether the input four letters representing moving directions are valid
    (i) Check whether the input is four letters separated by a space
    (ii) Check whether the input is digits
    (iii) Check whether the four letters are totally different from each other
(2) If the input satisfies all conditions above, the input is valid, return True.

4.choose_puzzle_nine_direction_and_move(p_direction):
Parameter: p_direction: the input letters representing moving directions
Usage:
(1) Show the valid sliding moves with the designated letter
(2) Move the sliding blocks according to user's instruction
(3) Show the updated puzzle on the screen after each move

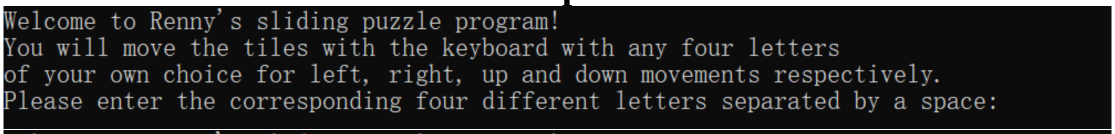
5. choose_puzzle_sixteen_direction_and_move(p_direction):
Parameter: p_direction: the input letters representing moving directions
Usage:
(1) Show the valid sliding moves with the designated letter
(2) Move the sliding blocks according to user's instruction
(3) Show the updated puzzle on the screen after each move

6.check_result_nine():
Parameter: no parameter
Usage: check whether the 8-tile puzzle is solved

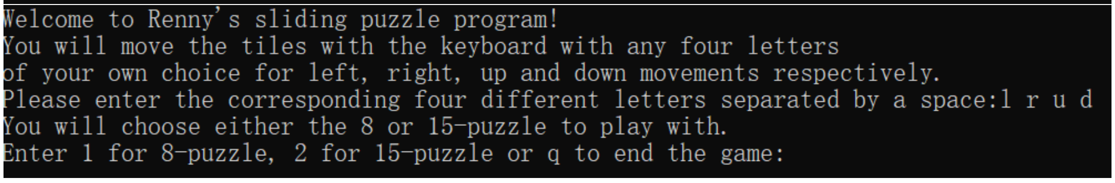
7. check_result_sixteen():
Parameter: no parameter
Usage: check whether the 15-tile puzzle is solved

8.main()
Parameter: no parameter
Usage:
(1) Conclude all the above functions
(2) Display a brief introduction about the game at the beginning
(3) Prompt the user to enter four letters for four direction move
(4) Prompt user for selection of either 8 or 15 puzzle
(5) Track total number of moves
(6) Execute the corresponding functions according to the user's instructions
(7) After the puzzle is solved, prompt up the user to continue another game or end the program.

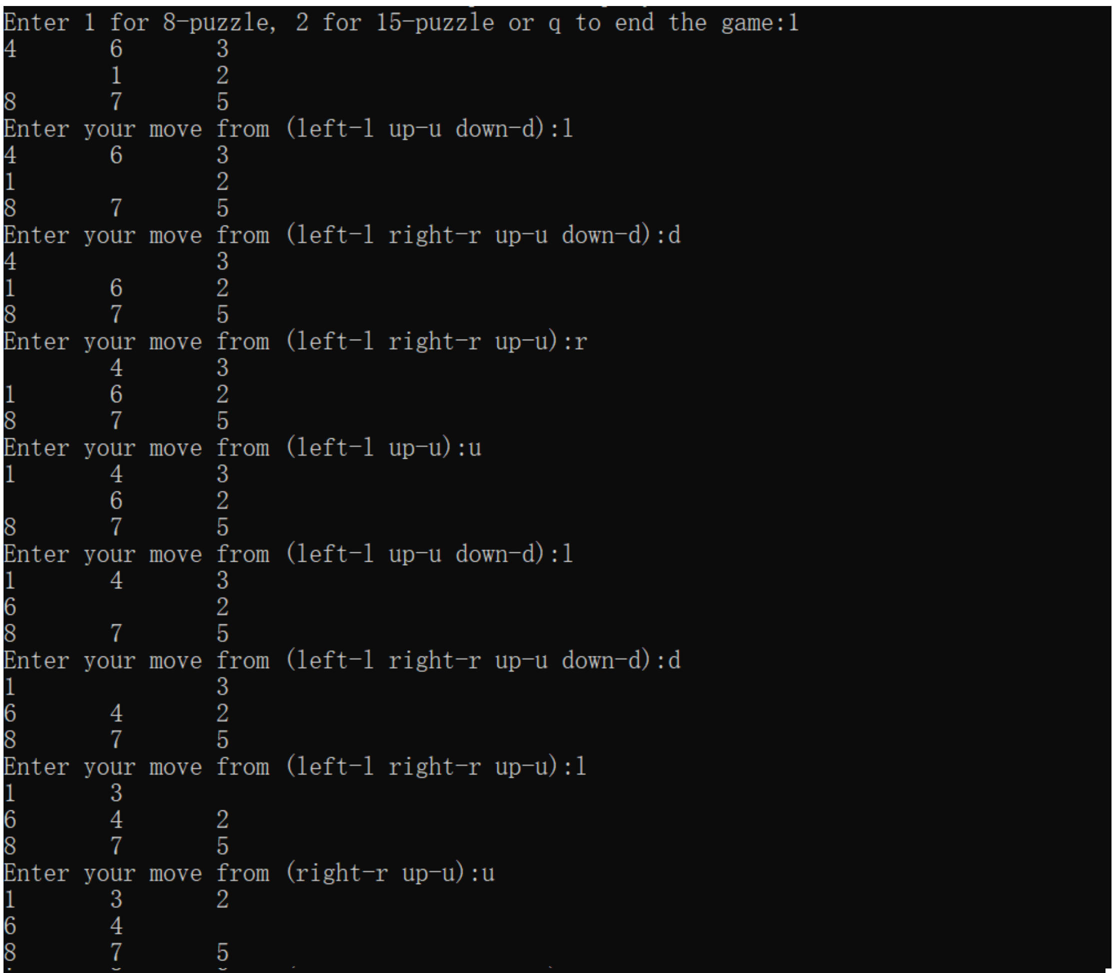
Output



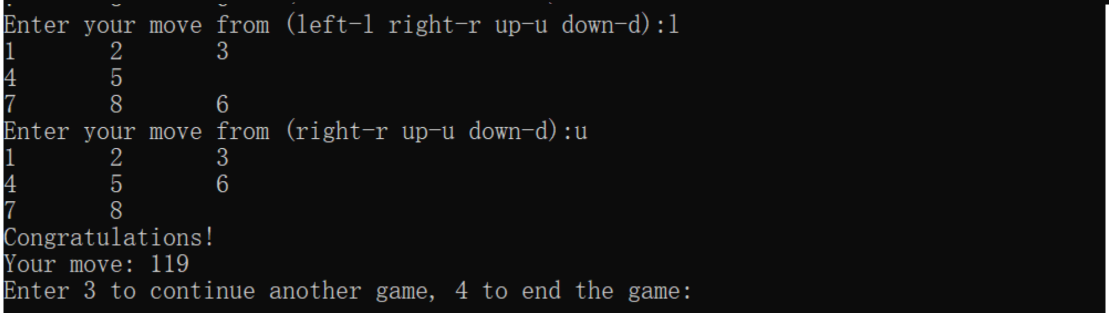
```
Welcome to Renny's sliding puzzle program!
You will move the tiles with the keyboard with any four letters
of your own choice for left, right, up and down movements respectively.
Please enter the corresponding four different letters separated by a space:
```



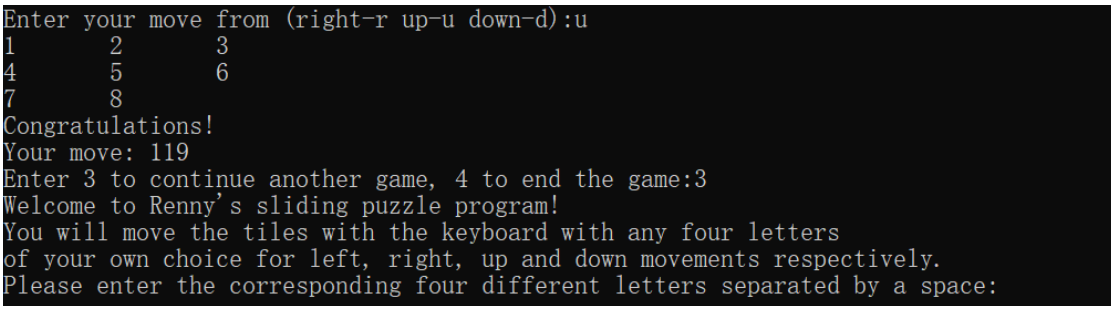
```
Welcome to Renny's sliding puzzle program!
You will move the tiles with the keyboard with any four letters
of your own choice for left, right, up and down movements respectively.
Please enter the corresponding four different letters separated by a space:l r u d
You will choose either the 8 or 15-puzzle to play with.
Enter 1 for 8-puzzle, 2 for 15-puzzle or q to end the game:
```



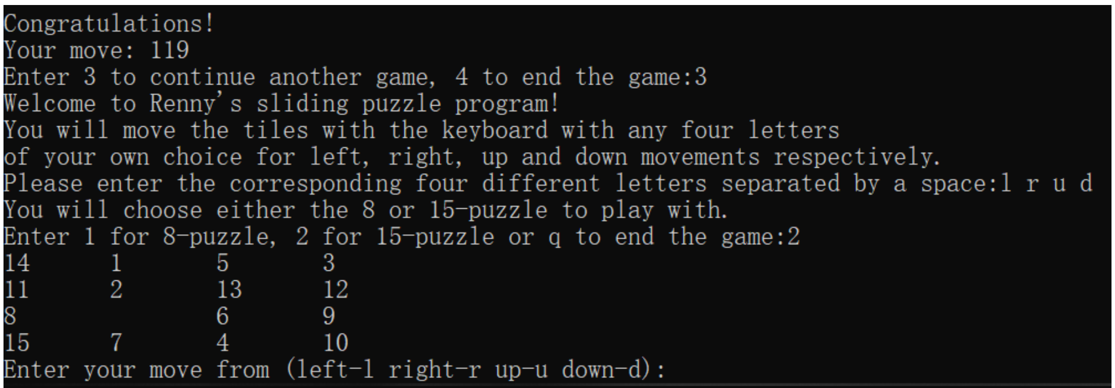
```
Enter 1 for 8-puzzle, 2 for 15-puzzle or q to end the game:1
4       6       3
        1       2
8       7       5
Enter your move from (left-l up-u down-d):l
4       6       3
1               2
8       7       5
Enter your move from (left-l right-r up-u down-d):d
4               3
1       6       2
8       7       5
Enter your move from (left-l right-r up-u):r
        4       3
1       6       2
8       7       5
Enter your move from (left-l up-u):u
1       4       3
        6       2
8       7       5
Enter your move from (left-l up-u down-d):l
1       4       3
6               2
8       7       5
Enter your move from (left-l right-r up-u down-d):d
1               3
6       4       2
8       7       5
Enter your move from (left-l right-r up-u):l
1       3
6       4       2
8       7       5
Enter your move from (right-r up-u):u
1       3       2
6       4
8       7       5
```



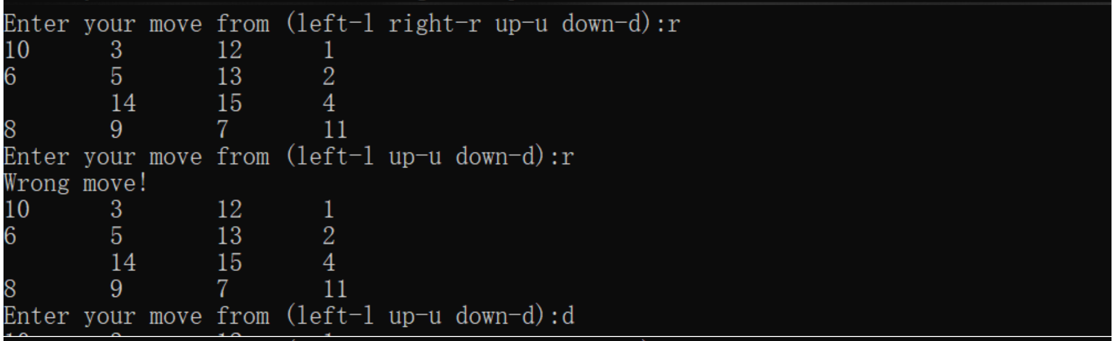
```
Enter your move from (left-l right-r up-u down-d):l
1       2       3
4       5
7       8       6
Enter your move from (right-r up-u down-d):u
1       2       3
4       5       6
7       8
Congratulations!
Your move: 119
Enter 3 to continue another game, 4 to end the game:
```



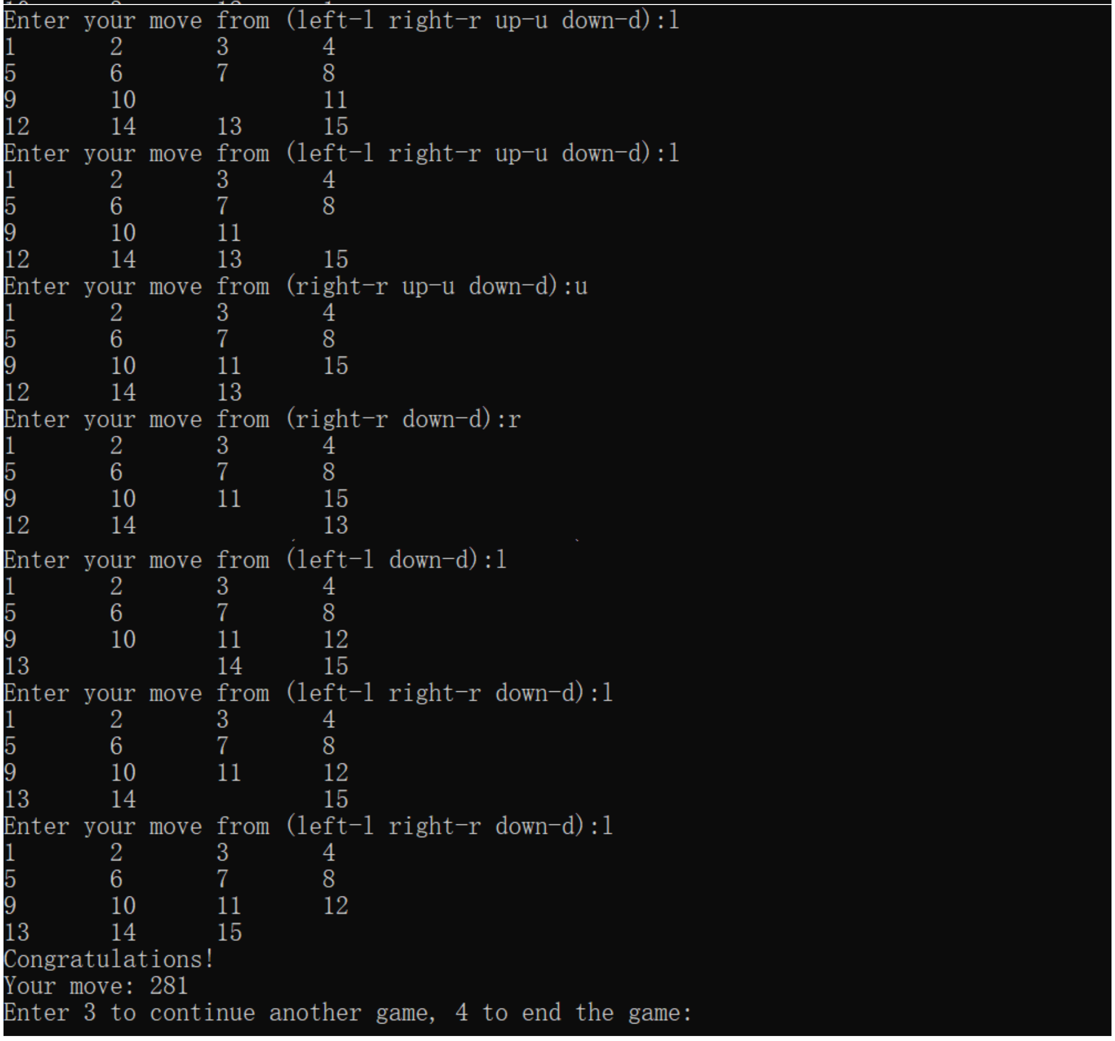
```
Enter your move from (right-r up-u down-d):u
1       2       3
4       5       6
7       8
Congratulations!
Your move: 119
Enter 3 to continue another game, 4 to end the game:3
Welcome to Renny's sliding puzzle program!
You will move the tiles with the keyboard with any four letters
of your own choice for left, right, up and down movements respectively.
Please enter the corresponding four different letters separated by a space:
```



```
Congratulations!
Your move: 119
Enter 3 to continue another game, 4 to end the game:3
Welcome to Renny's sliding puzzle program!
You will move the tiles with the keyboard with any four letters
of your own choice for left, right, up and down movements respectively.
Please enter the corresponding four different letters separated by a space:l r u d
You will choose either the 8 or 15-puzzle to play with.
Enter 1 for 8-puzzle, 2 for 15-puzzle or q to end the game:2
14      1       5       3
11      2       13      12
8               6       9
15      7       4       10
Enter your move from (left-l right-r up-u down-d):
```



```
Enter your move from (left-l right-r up-u down-d):r
10      3       12      1
6       5       13      2
        14      15      4
8       9       7       11
Enter your move from (left-l up-u down-d):r
Wrong move!
10      3       12      1
6       5       13      2
        14      15      4
8       9       7       11
Enter your move from (left-l up-u down-d):d
```



```
Enter your move from (left-l right-r up-u down-d):l
1       2       3       4
5       6       7       8
9       10              11
12      14      13      15
Enter your move from (left-l right-r up-u down-d):l
1       2       3       4
5       6       7       8
9       10      11
12      14      13      15
Enter your move from (right-r up-u down-d):u
1       2       3       4
5       6       7       8
9       10      11      15
12      14      13
Enter your move from (right-r down-d):r
1       2       3       4
5       6       7       8
9       10      11      15
12      14              13
Enter your move from (left-l down-d):l
1       2       3       4
5       6       7       8
9       10      11      12
13              14      15
Enter your move from (left-l right-r down-d):l
1       2       3       4
5       6       7       8
9       10      11      12
13      14              15
Enter your move from (left-l right-r down-d):l
1       2       3       4
5       6       7       8
9       10      11      12
13      14      15
Congratulations!
Your move: 281
Enter 3 to continue another game, 4 to end the game:
```