# CS 340 - Software Design

### Spring 2013

# Project 4 - Fifteen Puzzle

### Due: Thursday, April 18, 2013 at 11:59 pm

Students are allowed to work on this project with one other person. Only submit the project once for the group! Make sure that the names and NetIDs of both students are clearly identified in the README for the project. Both students will receive the same grade for the project regardless how the team divides the work. Note that part of the "extra work" for having a partner is to divide the work fairly. Some students may decide to work alone rather than depend on another student. The choice is yours. Choose wisely.

The Fifteen Puzzle is one variation of the N-Puzzle. The Fifteen Puzzle was made famous by Sam Lloyd who in 1878 offered $1000 to anyone who could solve an unsolvable variation of the puzzle. The puzzle consists of a 4x4 grid with the numbers from 1 to 15 on the grid leaving one grid position empty. The numbers are mixed up the puzzle is solved when the number are put into the following order:

| | | | |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | |

To solve the puzzle, a number that is next to the empty position is moved into the empty position. By "next to", the number can be above, below, to the left or to the right of the empty position. The empty position will now occupy the grid position were the number had been.

Your program is to implement the Fifteen Puzzle using the Qt Library GUI elements. One method to do this is to create a 4x4 grid of 16 buttons. Use the text of the buttons to specify which number (or the blank) is stored at that grid position. When the user clicks on a button that is next to the "blank" button, the text on those two buttons is exchanged. Clicking on any other button does nothing.

Your program will need a menu that will perform the following operations.

- Quit the program

  This menu button is in addition the quitting your program by clicking on the X on the right side of the title bar.

- Display an About box

  The About box can be a simple dialog box that gives the name of the program's author, when the program was written and why ("The 4th programming assignment for CS 340").

- Provide help on how to use your program.

  This can be a dialog box that describes the basics of your program. This should include a description of all menu operations.

- Reset the numbers in the grid.

  This should put all of the numbers into the "solved" ordering.

- Mix the numbers in the grid.

  This should rearrange the numbers in the grid into some random but solvable order.

A discussion of what makes an ordering of the numbers in the puzzle solvable or not can be found at http://mathworld.wolfram.com/15Puzzle.html. The basic idea is whether there are an odd or even number of permutation inversions compared to the solved ordering of the numbers. Any arrangement that has an even number of permutation inversions is solvable, while any arrangement that has an odd number of permutation inversions is unsolvable.

The easiest way to determine if an arrangement is solvable or not is to count the inversions made for each position. First think of the 4x4 grid as a one dimensional array were the position in the array is (row-1)*4 + col. Thus the positions of the grid as:

| | | | |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |
| 13 | 14 | 15 | 16 |

correspond to the array:

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |

For each value in the array, count the number of values in a higher position with a smaller value. This will give the number of permutation inversions. If this number is even, the puzzle is solvable. If this number is odd, the puzzle is not solvable.

The following discussion is taken from http://mathworld.wolfram.com/15Puzzle.html. Consider the following puzzle.

| | | | |
|---|---|---|---|
| 13 | 10 | 11 | 6 |
| 5 | 7 | 4 | 8 |
| 1 | 12 | 14 | 9 |
| 3 | 15 | 2 | |

The number 13 is at position 1. Since there are 12 values less than 13 at position higher than 1, the inversion count for the 13 is 12. The number 7 is at position 6. Since there are 4 values less than 7 at positions higher than 6 (the 4 at position 7, the 1 at position 9, the 3 at position 13 and the 2 at position 15), the inversion count for the 7 is 4. For the entire puzzle, the inversion counts are 12, 9, 9, 5, 4, 4, 3, 3, 0, 3, 3, 2, 1, 1, and 0, giving the total inversion count of 59. Since this number is odd, the above arrangement of the puzzle cannot be solved. Note that this example has the blank at position 16 and thus it is excluded from the inversion count. It may be simpliest to follow this idea and only randomize the numeric values on the grid.

## Extra Credit

You may earn 10 pts additional credit by having an image divided into 16 parts and displaying 15 of those parts instead of using the numbers from 1 to 15. Once the problem is solved (you will need to automatically detect this), display the 16th part of the image. It is your responsibility to get and prepare the image for your program.

## Submission of the Program

Submit your program via the assignment link for Project 4 in Blackboard. If you are working with a partner, only submit one copy of the assignment. Make sure that the names and NetIDs of both students are clearly identified in the README for the project.