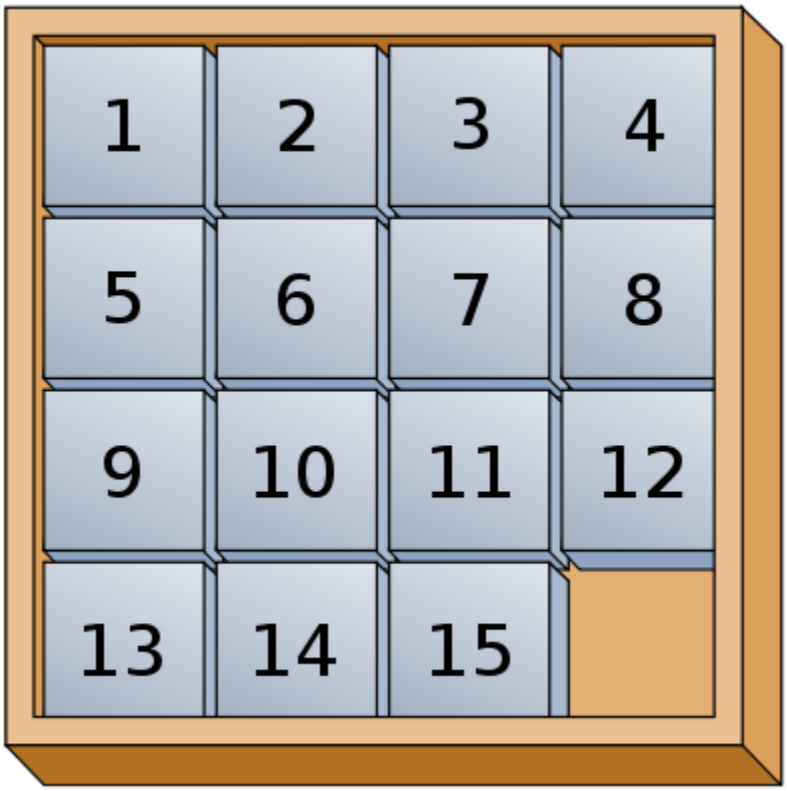
Due Date: March 26th, 2010

Program 3: Game FIFTEEN

You are facing a challenge! It is the Game of Fifteen! Also known as the FIFTEEN puzzle, this game consists of a 4 x 4 board filled with fifteen tiles numbered one through fifteen. There is an empty square that remains. At any one time you can slide an adjacent tile into the empty square to modify the tile layout on the playing field. The goal of the game is to solve a scrambled puzzle, by bringing it to its initial position. The initial position is below.



Deliverables:

- 3-page paper
- FIFTEEN program base
- Animation of the puzzle (2% bonus)
- Animated solving of the puzzle (3% bonus)
- (for 2 people) A statement of work describing who did what parts of the project

Paper

Your paper should describe the project. The paper is to be concise yet descriptive. You may include things like your ideas behind the GUI interface, your algorithm of animating the tiles, solving the puzzle, etc. etc. The paper is to include at least one UML diagram showing class composition of your project, including description of UML diagram parts. You may include images, prototypes, and anything else that will help me or a potential customer understand your project.

Due Date: March 26th, 2010

FIFTEEN Program Base

Upon running, the program should display the graphical (GUI) interface showing the FIFTEEN game puzzle field with the fifteen tiles in their initial position. Clicking on a tile should move it to a nearby empty spot if one is available. The tile move does not have to be animated.

Your program should have a Scramble button. Clicking the button should scramble your puzzle. You can display the puzzle in a scrambled state after clicking the button. You do not have to visually show the scrambling process. Note, however, that the final solved state (which is the same as the initial state) is not reachable from all puzzle configurations. This means you will have to scramble the puzzle by moving the pieces from the initial state. You cannot simply generate a scrambled state randomly, as there may not be a solution for it.

Write your program in such a way that it can be run as an Applet and as a program. See Section 17.5, p. 567 on how to do this.

Bonus 1– Animation

This is a 2% bonus. Animate the tile pieces as they move. Clicking the Scramble button should scramble the puzzle by animating the tile buttons.

Bonus 2– Animated Solving of the Puzzle

This is a 3% bonus. Include a Solve button in your program. Clicking the Solve button should visually solve the puzzle from any possible solvable state. Solution must be reached in a reasonable amount of time (i.e. under 1 minute)

Grading

- 70% - program
- 30% - paper
- 2% - bonus 1
- 3% - bonus 2

Other

- You can work in groups of 2. Be clear on who does what. You may find it better to split the tasks between you ahead of time, based on your experience and expertise.