YOU MUST DEMONSTRATE YOUR GAME DURING CLASS TIME.

Assignment due: see eConestoga

Before you start:

If you are not familiar with 15-puzzle game, please check the following links:

1. <http://en.wikipedia.org/wiki/15_puzzle>
2. <http://migo.sixbit.org/puzzles/fifteen/>

Self Study:

1. Saving a text file and loading from a text file
2. Using OpenFileDialog and SaveFileDialog controls

Check FileApplication.zip application available under Week 5 in D2L.

Problem Specification: (15-Puzzle)

1. Create a 15-Puzzle game in C#.
2. You may use custom-controls/user-controls for creating the game.
3. The game will show game-end message after solving the puzzle.
4. The game will be initialized randomly.
5. The game will have a save game and a load game option (you may use menu or buttons).
6. Save game option will save the current game state to a text file.
7. Load game option will let the user to load an existing game file.
8. You must use OpenFileDialog and SaveFileDialog controls to load and save. Use C:\ as the default path for save file and load file.
9. After ending the game, initialize it randomly so that another game can be played.
10. Enhance the game so that it can be customized to an n-puzzle by providing number of rows and columns value through text boxes.

**Remember to incorporate the followings for all assignments (if applicable)**

* Add Title comment.
* Add Documentation comment.
* Add Implementation comment where (you think) necessary.
* All other standards.
* Organize the controls to have a better look.

Note: Use your own imagination for designing the user interface. The snapshot of the Form is given only to clarify the requirement. You don’t have to make it exactly the same.

Hints:

* You must understand that 50% of the randomly generated combinations are not solvable. That’s why, while testing your program you should use a valid initial combination (you may use the combination shown in the screen shot, or generate one by your own). Please add 2 game files (valid solvable combination file for 15-puzzle) in the zip file that you upload.
* When you are initializing your game with random combination, don’t worry about the solvability. Whether your game can identify end game condition will only be tested with the valid combinations (2 save games) you provide.
* While saving the combination you may save the number of rows and number of columns as well, in the file, so that you can recreate the maze.
* Test for 15-Puzzle whether it can identify game end state. Since, for n-puzzle there can be numerous combinations, you don’t need to test it rigorously.
* Optional (no extra mark): if you always want to generate solvable combination, there are algorithms available to test whether a combination is solvable. You might also start with a game end combination (1 to n) and then scramble it randomly moving the empty slot, say 100 times.

**Hardcopy Submission Requirements (follow the sequence as stated)**

1. Combined Coverpage and Marking sheet.
2. Assignment 3 Marking sheet. (last page of this document)
3. Run your program, and then take a snapshot (printscreen your form).
4. Printout of your code. Print using Visual Studio 2013 IDE.
5. Make sure all Programming standards are followed. Read pages 12-40 for detailed information : [*http://www.conestogac.on.ca/cpa/common/handbook/Standards.pdf*](http://www.conestogac.on.ca/cpa/common/handbook/Standards.pdf)
6. Make sure all Assignment standards are followed.

Note: Repeated violation of the same standard is counted.

Softcopy Submission Requirements

1. Make your solution folder a zip file according to the following example:

if the students’ names are Jason Bourne, and Indiana Jones for

Assignment 3, the name of the project will be JBIJAssignment3 and the zip file

will be JBIJAssignment3.zip

1. Log in to your D2L account.
2. Select the course PROG2370 and locate the Assignment’s dropbox
3. Upload the zip file. (

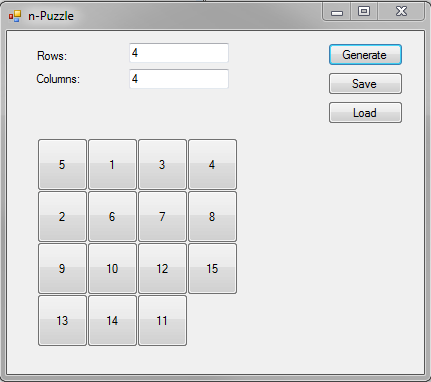


Figure 1: n-Puzzle screenshot

Assignment 3 Marking Sheet

**Name:\_\_\_\_\_Yoonsuk Cho\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(please fill your name)**

|  |  |
| --- | --- |
| **Spec** | **Marks** |
| (Attractive/Colorful) User Interface Design | /05 |
| Game starts with a randomly combination (15 tiles) | /10 |
| Game can be loaded from file | /15 |
| Game can be saved | /15 |
| Game shows Game End message at game end. | /15 |
| When one game completes, it initializes to a random combination – ready to be played again(don’t worry about its solvability) | /10 |
| Game is customizable to an n-puzzle (r x c rows and columns combination) through TextBoxes. | /20 |
| Using OpenFileDialog and SaveFileDialog controls | /05 |
| 2 solvable combinations as text files (15 puzzle only) provided in the zip file. | /05 |
| **Total** | **\_\_\_\_\_\_\_\_\_\_\_\_/100** |

**Deduction:**

|  |  |
| --- | --- |
| Runtime errors | 15 x \_\_\_\_\_\_\_\_\_\_\_\_ = |
| Standard | 5 x \_\_\_\_\_\_\_\_\_\_\_\_ =\_\_\_\_\_\_\_\_/20 |
| Programming Standard | 1 x \_\_\_\_\_\_\_\_\_\_\_\_ =\_\_\_\_\_\_\_\_/20 |
| Late Submission | 20 x\_\_\_\_\_\_\_\_\_\_\_\_\_= |
| **Total Deduction** |  |

|  |  |
| --- | --- |
| **Total Marks** |  |