**Documentation and testing for software development test – Lights Out**

**Objective**

Lights Out is a puzzle game consisting of an n x n grid of lights. At the beginning of the game, some of the lights are switched on. When a light is pressed, this light and the four adjacent lights are toggled, i.e., they are switched on if they were off, and switched off otherwise. The purpose of the game is to switch all the lights off.

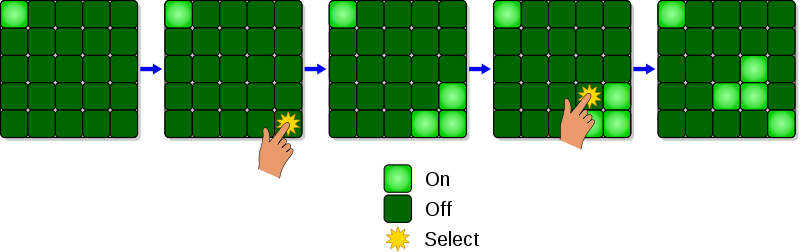


Image from Wikipedia.

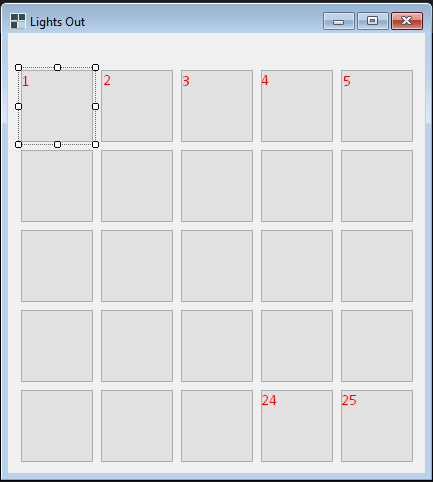
Implement the game for a 5 by 5 grid in the .Net environment.

The game must start with some lights turned on, and the user keeps playing till he either gives up or turns off all the lights.

**Plan of action**

1. Design a windows form for a user to interact with.
2. Create an algorithm for the game play.
3. Development, coding, and form creation.
4. Testing and further development if necessary.
5. **Design**

* form with 25 buttons in a 5 by 5 grid
* buttons added in a numerical order



1. **Algorithm**

* Inform player about the rules of the game;
* Start the game with randomly switched on buttons;
* Action – button click:
  + Get adjacent buttons
  + Change the colour of the affected buttons
  + Check if the user has won:
    - If won – end the game > start a new game
    - Else – wait for next button click

1. **Development**
   1. Message box to inform user about the rules of the game. *(note: message box to be shown after the game/form has been initialised)*
   2. Handling of a button click; declaration of variables necessary to hold the data about current state of the buttons. *(note: additional variable required to hold buttons that are to be changed upon the click action)*
   3. Functions for:
      1. Getting adjacent buttons – switch case for each of the button based on the button’s name (position); *(note: using switch case would require manual coding for each of the buttons; mathematical formula created to find adjacent buttons, which could be reused in case the grid’s size was amended)*
      2. Update affected buttons’ values and change their colours; *(note: split into two different functions to simplify the processing and ensure that variables are handled correctly)*
      3. Check if the game was won. *(note: check to happen upon the change of colours)*
2. **Testing**
   1. Alpha testing, results and solutions
      1. Test of all the buttons to check if the functions are being executed as expected

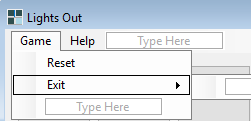
Result: clicking buttons at the beginning and end of the rows would change buttons that were subsequently numbered, even if they were not adjacent; i.e. clicking on the button number 5, would change the colour of buttons 4. 10 and 6.

Solution: used modulus function to identify which buttons are at the beginning and the end of the row to exclude them from the list of buttons to be modified.

* + 1. Test of ‘game won’ scenario – change colour of the buttons to green, inform the player that they have won, and reset the game.

Result: Due to the randomness of the initial values of the buttons, added test code to check if it works. Winning of the game executed as expected.

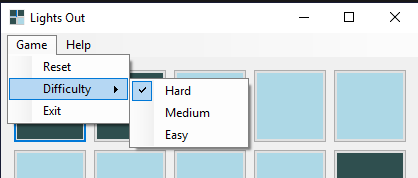
Solution: added a menu strip with two options: Reset, Exit.





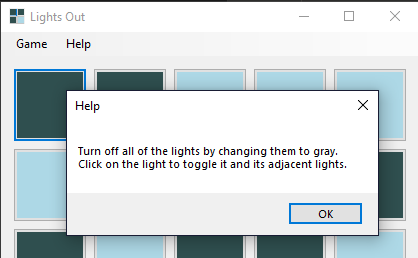
* 1. Beta user testing, comments and changes
     1. The user loses interest if it’s too hard to win the game.

Change: Added difficulty levels – new variable to store level of difficulty as picked by the player; minor changes in code to amend winning conditions; new item in the menu to pick the difficulty level. By default, game is started in a hard mode (all lights out) – as per initial project specifications.



* + 1. Message box upon the start of the game is unnecessary.

Change: Removed the message box and created a new menu item – Help, which will provide the user with instructions on how to play the game.



**Post-testing**

The whole solution was recreated under a new name (original name Game Off) to match the specifications. Renamed a few of the variables to retain the naming standard across the whole project. Tidied up inconsistencies in names of event handlers.