Your task is to implement the simple memory game according to functional requirements and your best knowledge.

Source code must be located in the github repository. If you don't have a github account, create one at https://github.com/. During game development **work with git according to your best known practices**.

Write your solution in C#. You have 4 days to deliver a given application. When the task is finished please send a github link as a reply.

Functional requirements were delivered by the client.

1. Program should load a text file with words that will be covered and doubled during the game. There is a file attached "words.txt", which you need to use.
2. Program should ask the user to choose between two difficulty levels. Easy - program loads 4 **randomly selected** words to discover and player has 10 chances to reveal all memory, or Hard - user needs to guess location of 8 **randomly selected** word pairs and has 15 chances to reveal all word pairs.
3. After difficulty level pick, User gets displayed matrix of covered words ex.

—----------

Level: easy
Guess chances: 10

1 2 3 4
A X X X X
B X X X X

—----------

Then the program asks the user to choose the first word to uncover, by coordinates ex. **A1**. which triggers the logic responsible for uncovering words.

After the player enters coordinates, the program should display a matrix with an uncovered single word.

—----------

Level: easy
Guess chances: 10

1 2 3 4
A Tomato X X X
B X X X X

—----------

Next, the player should be prompted to enter second coordinates ex. **B4**. If the words match, words stay uncovered and the gameplay continues until the player loses all chances or unreveal all words. If the words don't match, the words get covered again and the player loses one guess chance.

4. Add a question about restarting the program after wins or losses.
5. **OPTIONAL** Add information about chances taken and guessing time at the end of the game (i.e. "You solved the memory game after 8 chances. It took you 240 seconds").
6. **OPTIONAL** Add a high score - some people take pride in their score. At the end of a successful game program should ask the user for his/her name and save that information to a file - name| date | guessing_time | guessing_tries |
7. **OPTIONAL** Expand high score - program should remember 10 best scores (read from and write to file) and display them at the end, after success/failure.
8. **OPTIONAL** Beautify your game! Add ASCII art, improve word matrix, or anything to make your program more appealing to the player!

Technical annotation: Utilize as much clean code and object-oriented good practices as possible. Remember to ensure your solution stability - make sure that the user won't encounter any critical errors or exceptions that are likely to stop a program!

In case you are not familiar with the concept of memory game - you can check out its description on Wikipedia - Memory game or play one in the browser to visualize better gameplay - Example web memory game

Remember - there is no one perfect solution for the application.

If you reach this far - please add to your Readme.MD just a few sentences on why do you code - is it just a professional matter or something more? We would like to know the answer!

Good luck!
Motorola Academy team