# Refactoring Documentation for Project “Hangman-5”

Author: Team-Samarium

1. Reformatted the source code:
   * Removed all unneeded empty lines.
   * Removed unused usings.
   * Added “this” where needed.
   * Added class and method access modifiers.
   * Inserted empty lines between the methods.
   * Formatted the curly braces **{** and **}** according to the best practices for the C# language.
   * Put **{** and **}** after all conditionals and loops (when missing).
   * Character casing: variables and fields made **camelCase**; types and methods made **PascalCase**.
2. Renamed variables and methods:
   * In class **Scoreboard:** TopFiveRecords -> highScoreList
   * In class **Scoreboard:** MAX\_NUMBER\_OF\_RECORDS-> MaxNumberOfHighScoreEntries
   * In class **Scoreboard:** scoreQualifiesForTopFive-> scoreQualifiesForHighScoreList
   * In class **Scoreboard**: CheckIfQualifiesForTopFive() -> CheckIfQualifiesForHighScoreList()
   * In class **Scoreboard**: CompareByKeys() -> CompareByValue()
   * In class **Scoreboard**: CheckIfQualifiesForTopFive() -> CheckIfQualifiesForHighScoreList()
   * In class **Hangman:** PlayOneGame() 🡪 NewGame()
   * In class **Hangman:** ff 🡪 changed to field isGameWon
   * In class **Hangman:** ff2 🡪 field isHelpUsed
3. Changes in **Scoreboard**:
   * **highScoreList:** List<KeyValuePair<int, string>> 🡪 List<KeyValuePair<string, int>>
   * **CompareByValue:** return pairA.Key...(pairB.Key) 🡪pairA.Value..(pairB.Value)
   * **PrintCurrentScoreboard:** name 🡪 changed Value to Key.
   * **PrintCurrentScoreboard:** mistakes 🡪 changed Key to Value.
   * Added **ToString().**
   * Added a **parameter** to the **constructor** specifying how many games should be played
   * Removed some unnecessary code.
4. Changes in **Hangman**:
   * **HelpByRevealingALetter():** extracted to **DisplayUtilities**
   * Changed the **constructor** so that it takes the number of games to be played
   * **PrintWelcomeMessage():** extracted to **DisplayUtilities**
   * **PrintInvalidEntryMessage():** extracted to **DisplayUtilities**
   * **PrintDisplayableWord():** extracted to **DisplayUtilities**
   * **SelectRandomWord():** extracted to **WordUtilities**
   * **GenerateEmptyWordOfUnderscores():** extracted to **WordUtilities**
   * **CheckIfWordIsRevealed():** extracted to **WordUtilities**
   * **CheckUserGuess():** extracted to **WordUtilities**
   * **CheckIfLetterIsAlreadyRevealed():** extracted to **WordUtilities**
   * Added field **isCurrentGameOver** .
   * Added field **isGameWon.**
   * Added field **isWholeGameOver.**
   * Added field **isHelpUsed.**
   * **Main():** extracted to **MainClass.**
   * Created new method **Play():** containing logic from **Main().**
   * Created new method **GetUserInput().**
   * Created new method **GetInputType().**
   * Removed some unnecessary code.
5. Createdclass **DisplayUtilities** that abstracts the visualisation logic.
   * Added method **DisplayMessage** so that the code is more flexible in case we need to change the output from console to GUI.

* Method **RevealALetter** that reveals the next unrevealed letter
  + Methods **PrintWelcomeMessage** and **PrintInvalidEntryMessage** that display the relevant messages
  + Method **PrintDisplayableWord** that prints the word in the way it is supposed to be displayed

1. Created class **WordUtilities** that abstracts word operations**.**

* Method **SelectRandomWord** that picks a random word out of an array
* Method **GenerateEmptyWordOfUnderscores** that generates the underscore representation of a word with a given length
* Method **CheckIfWordIsRevealed** that checks whether all the letters are uncovered
* Method **CheckUserGuess** that checks whether the guess is contained in the unrevealed letters
* Method **CheckIfLetterIsAlreadyRevealed** that checks whether a letter has already been revealed in a word

1. Created enum **InputType**.
2. Added UnitTests.

* 45 unit tests total
* Used both helper heir classes and Console.SetIn and Console.SetOut to implement the tests
* 96% covarage according to dotCover