Hangman Game Documentation

Written by: Nofal Adnan Butt

Date: 03-07-2022

# Requirements:

As part of the assignment given by ING, a hangman game had to be made with the following requirements:

1. Single player game against computer.  
2. Computer will choose a word and the player will try to guess.  
3. Until the player wants to quit, they should be able to continue with a new round.

The basic rules of the game can be seen at:

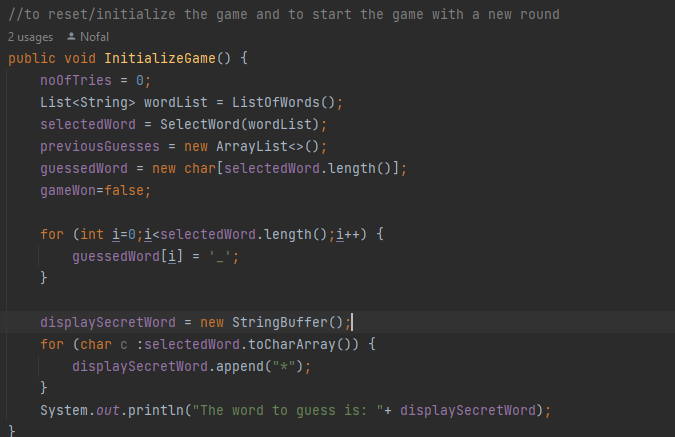
<https://en.wikipedia.org/wiki/Hangman_(game)>

# Code and implementation:

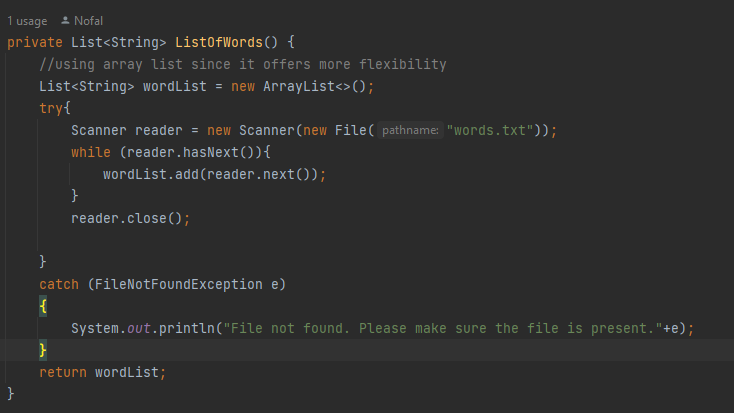
The game has been written in Java programming language using java 18. The code for the game can be found on GitHub using this link:

<https://github.com/NofalB/Hangman>

The main source code contains two main classes. One class is the main class of the program while the other class is called Hangman. The class Hangman contains all the code required for the Hangman game to work properly. The class **InitializeGame()** is used to reset/initialize the game and to prepare the game again if the user wants to play again after winning/losing.



**ListofWords()** method is called to get a list of words from the file that has been made a part of source code. These words are selected randomly each time the game is run or started.



The **PlayHangman()** function contains the main logic required to play the hangman game. The whole game is running wrapped in a while loop where the game keeps running until the user wants to quit. There are multiple checks within this loop. The first if statement checks if the no of tries done by the user do not exceed the max number of tries which is a constant specified in the code and that the game has not been won. A round is done when the user either wins or loses the game and is then given the option to either start a new round or exit the game. There are also multiple functions called within this method like a method to print the current guess so far(**PrintCurrentGuess()**), **ValidateUserInput()** to make sure that whatever user entered is a valid character and regex is used to verify if the character entered by the user is indeed an alphabet as well as a method called **IsGameWon()** to check if the guessed word matches the selected word by the computer. There is also a check to make sure that the entered character by the user has not been guessed before, if that is the case, the user is informed and the try does not count. The code snippet for this **PlayHangman()** function can be seen below:



The **Main** method of this application can also be seen below:

