Object Oriented Programming with Java

REPORT ON

HANGMAN GAME

PREPARED BY

TEJASHWINI.J

USN-1CD23CS181

PROF-SHILPA S.B.

Definition :

**Hangman** is a [guessing game](https://en.wikipedia.org/wiki/Guessing_game) for two or more players. One player thinks of a [word](https://en.wikipedia.org/wiki/Word), [phrase](https://en.wikipedia.org/wiki/Phrase), or [sentence](https://en.wikipedia.org/wiki/Sentence_(linguistics)) and the other(s) tries to guess it by suggesting [letters](https://en.wikipedia.org/wiki/Letter_(alphabet)) or [numbers](https://en.wikipedia.org/wiki/Number) within a certain number of guesses. Originally a [paper-and-pencil game](https://en.wikipedia.org/wiki/Paper-and-pencil_game), there are now electronic versions.

**Overview :**

The word to guess is represented by a row of [dashes](https://en.wikipedia.org/wiki/Dash) representing each [letter](https://en.wikipedia.org/wiki/Letter_(alphabet)) or [number](https://en.wikipedia.org/wiki/Number) of the [word](https://en.wikipedia.org/wiki/Word). Rules may permit or forbid [proper nouns](https://en.wikipedia.org/wiki/Proper_noun) (such as names, places, or brands) or other types of words (such as [slang](https://en.wikipedia.org/wiki/Slang)). If the guessing player suggests a letter which occurs in the word, the other player writes it in all its correct positions. If the suggested letter does not occur in the word, the other player adds (or alternatively, removes) one element of a hanged [stick figure](https://en.wikipedia.org/wiki/Stick_figure) as a [tally mark](https://en.wikipedia.org/wiki/Tally_marks). Generally, the game ends once the word is guessed, or if the stick figure is complete—signifying that all guesses have been used.

The player guessing the word may, at any time, attempt to guess the whole word. If the word is correct, the game is over and the guesser wins. Otherwise, the other player may choose to penalize the guesser by adding an element to the diagram. If the guesser makes enough incorrect guesses to allow the other player to complete the diagram, the guesser loses. However, the guesser can also win by guessing all the letters that appear in the word, thereby completing the word, before the diagram is completed.

Code :

import java.util.Scanner;

import java.util.Random;

public class HangmanGame {

private static final String[] WORDS = {"java", "hangman", "programming", "computer", "developer", "software", "android"};

private static String wordToGuess;

private static StringBuilder currentGuess;

private static int attemptsRemaining;

private static boolean[] guessedLetters;

public static void main(String[] args) {

// Initialize the game

Scanner scanner = new Scanner(System.in);

Random random = new Random();

wordToGuess = WORDS[random.nextInt(WORDS.length)];

currentGuess = new StringBuilder("\_".repeat(wordToGuess.length()));

attemptsRemaining = 6; // Maximum number of incorrect attempts

guessedLetters = new boolean[26]; // Track guessed letters (a-z)

System.out.println("Welcome to Hangman!");

System.out.println("Try to guess the word.");

while (attemptsRemaining > 0 && currentGuess.toString().contains("\_")) {

System.out.println("\nCurrent word: " + currentGuess);

System.out.println("Attempts remaining: " + attemptsRemaining);

System.out.print("Enter a letter: ");

char guessedLetter = scanner.next().toLowerCase().charAt(0);

// Check if the letter is valid

if (guessedLetter < 'a' || guessedLetter > 'z') {

System.out.println("Please enter a valid letter (a-z).");

continue;

}

if (guessedLetters[guessedLetter - 'a']) {

System.out.println("You already guessed this letter.");

continue;

}

// Mark the letter as guessed

guessedLetters[guessedLetter - 'a'] = true;

// Check if the guessed letter is in the word

if (wordToGuess.indexOf(guessedLetter) != -1) {

// Reveal the guessed letter in the current guess

for (int i = 0; i < wordToGuess.length(); i++) {

if (wordToGuess.charAt(i) == guessedLetter) {

currentGuess.setCharAt(i, guessedLetter);

}

}

System.out.println("Good guess!");

} else {

// Wrong guess

attemptsRemaining--;

System.out.println("Wrong guess!");

}

}

if (currentGuess.toString().equals(wordToGuess)) {

System.out.println("\nCongratulations! You guessed the word: " + wordToGuess);

} else {

System.out.println("\nGame over! The word was: " + wordToGuess);

}

scanner.close();

}

}

Explanation:

1.Word List: The WORDS array contains a set of predefined words for the game.

2. Game State: wordToGuess: The word that the player needs to guess. guessedWord: A StringBuilder that holds the current state of the word with blanks for unguessed letters. guessedLetters: A set that tracks all the letters the player has guessed. attemptsLeft: Tracks the number of incorrect guesses allowed (6 in this case).

3. Game Loop: The game continues as long as there are remaining attempts. The program prompts the player to enter a letter and checks if it's correct. If the letter is correct, the guessed word is updated. If the letter is incorrect, the number of attempts left is reduced.

4. End Conditions: The game ends when the player guesses all the letters correctly or runs out of attempts.

How to Run:

\*Copy and paste the code into a Java file (e.g., HangmanGame.java).

\*Compile and run it in your IDE or command line using:

🡪javac HangmanGame.java

🡪java HangmanGame

Output :

Welcome to Hangman!

Try to guess the word.

Current word: \_\_\_\_\_\_\_

Attempts remaining: 6

Enter a letter: p

Wrong guess!

Current word: \_\_\_\_\_\_\_

Attempts remaining: 5

Enter a letter: j

Good guess!

Current word: j\_\_\_\_\_

Attempts remaining: 5

Enter a letter: a

Good guess!

Current word: ja\_\_\_

Attempts remaining: 5

Enter a letter: v

Good guess!

Current word: java\_

Attempts remaining: 5

Enter a letter: z

Wrong guess!

Congratulations! You guessed the word: java