PSDL MINI PROJECT : Hangman Game

BATCH : B1

Name : Prajakta Jadhav

Synopsis :

1) Problem Statement :

The program creates a word guessing game (Hangman Game).

2) Keywords :

A) Tkinter :

Tkinter is standard python library use for creating graphical user interfaces(GUI) .It provides a set of tools and widgets to build window,buttons,labels and other visual elements allowing developers to create interactive application with user friendly interface.

B) Import :

Used to bring external module and libraries to be used in code .

C) def :

Defines a function or method .

D) Global :

Declares global variable within function.

E) for:

Iterates over sequence .

F) TopLevel :

Create a new window within application.

G) Label/Button :

Widgets used in Tkinter library for graphical user interface elements.

H) PhotoImage :

Loads an image from file to be used within GUI.

I) Random:

Python Random Module is useful in implementing a randomization algorithm.

J) List :

List is the data structure that contain ordered collections of elements .List is mutable.

3) Abstract :

The code represents a simple Hangman game implemented using Python and Tkinter library for graphical user interface.It allows users to select category using Python and the Tkinter library for graphical user interface .It allows user to select category (animals,fruits or countries) and guess the hidden word by selecting letters .The player has limited number of attempts and there are hints available that revel a single letter frpm the word .The game features win and loose scenories or displaying pop for each.The code Structure includes functions for initializing the game,dispalying interfaces ,managing guesses and handling user interfaces.

***key features :

1) Graphical user interface :

Utilizes the tkinter library to create an interactive user interface with buttons ,labels and image for an engaging user experience.

2) Word Categories :

Offers multiple categories for users to select and play with adding variety to gameplay.

3) Game Mechanics:

Implements the classic Hangman game logic, allowing users to guess letters, display correct guesses, manage incorrect attempts, and keep track of guessed letters.

4) Limited Attempts:

Sets a limit on the number of incorrect attempts allowed, leading to a win or loss scenario based on successful word guessing within the attempts.

5) Hint System:

Provides the option for users to receive hints, revealing a single letter from the hidden word, with a restriction on the number of hints available.

6)Win/Lose Pop-ups :

Displays pop-up messages notifying the user when they win or lose the game, along with options to replay or exit.

7)Image Handling:

Integrates images, such as hangman progression images, to visually represent the player's progress and the consequence of incorrect guesses.

8)Replay and Exit Functionality:

Allows users to either restart the game or exit the application.

****Functions :

1) getword()

Purpose: Generates a random word from a provided word list.

Arguments: Takes a list of words (wlist) as an input.

Functionality: Utilizes the random module to select a word randomly from the provided word list and returns that word.

2) animal()

Purpose: Initiates the game with an animal-related word. *Functionality*: Selects a random word from the 'animal' category list and sets up the game environment to guess that word.
Resets: Clears any previous game data (such as guessed letters) and initializes new data for the selected word.

3) fruit()

Purpose: Begins the game with a fruit-related word.
Functionality: Chooses a random word from the 'fruit' category list and sets up the game environment for guessing that word.
Resets: Clears any previous game data and prepares new data for the chosen word

4) country()

Purpose: Starts the game with a country-related word.
Functionality: Picks a random word from the 'country' category list and sets up the game environment for guessing that word.
Resets: Clears any prior game data and initializes new data for the selected word.

5) callhint()

Purpose:

Provides a single hint by revealing a letter from the hidden word in the Hangman game.

Hint Mechanism:

Utilizes the random module to select a letter that has not been guessed yet. Reveals this letter to the player as a clue, assisting in the word-guessing process.
Limitations:
Allows a limited number of hints (two hints per word in this implementation).
Player Assistance:
Helps players by offering a small piece of information, improving their chances of guessing the word correctly.
Game Strategy:
Players strategically use hints to gain partial information about the word, assisting in making more accurate guesses and enhancing their chances of winning the game.

6)popup():

Purpose:
Initiates a separate window that contains the interactive gameplay elements for Hangman.
Gameplay Environment:
Creates a new window (Toplevel) specifically for the Hangman game. Includes graphical elements such as hangman image, word to guess, buttons for letter selection, attempts left, and a hint button. Display Management:
Manages the visual representation of the game elements, updating them according to the game progress.
Controls the display of hangman images based on the number of incorrect attempts remaining.
Player Interaction:
Enables player interaction through letter selection buttons, allowing guesses for the hidden word.
Allows players to click the hint button to receive hints during the game.
Outcome Display:
Shows pop-up messages indicating the win or loss scenarios, providing the player with feedback on the game result.

Game Loop:
The function loops until the game ends, offering an engaging and interactive environment for playing Hangman

4) Module-wise Description:

1) Tkinter:
Used for creating the game interface and managing GUI elements like buttons, labels, and windows.

2) Random:
Used for generating random words from predefined word lists for different Categories.

***Flow of the program :

1) Initialization:
The code starts by importing necessary libraries like random, tkinter, and PIL.Predefined word lists for different categories (animals, fruits, countries) are created.Variables for storing the chosen word, guessed letters, guessed word, and attempts are initialized.

2) Main Window Creation:
The main GUI window is created using Tkinter, displaying a welcome screen with buttons to start the game, access rules, and exit.

3) Game Initiation:
When the user clicks the start button, the choosecategory() function is called to present options for word categories.Upon selecting a category (e.g., animals), the corresponding function (animal(), fruit(), country()) is triggered.

4) Category Selection:
Upon selecting a category, a new window (Toplevel) opens, displaying buttons for different categories (e.g., animals, fruits, countries).

5) Gameplay Window:
After selecting a category, a new window (Toplevel) opens for gameplay. This window displays the hangman image, the word to guess (initially as underscores), buttons for letter selection, attempts left, and a hint button.

6) Game Mechanics:
Players can click the letter buttons to make guesses.
The checkletter() function manages the validation of the guessed letter. It updates the display, decreases attempts, and handles win/loss scenarios.
The display() function updates the game display, showing the guessed letters, attempts left, and hangman image based on incorrect attempts.

7) Hints:
Users can click the hint button (limited to two uses per word). The clickedhint() function controls the hint count and the callhint() function reveals a single letter from the word.

8) Win or Lose:
If the player guesses the entire word correctly, a pop-up window congratulates them on winning.
If the player exhausts all attempts without guessing the word, another pop-up window displays the loss along with the word and options to replay or exit.

9) Exit or Replay:
Users can choose to exit the application or replay the game, which loops back to the main window for category selection.

5)Technology, Science, and Features Covered:

Technology:

Python programming language, Tkinter library for GUI, PIL for image handling, random module for word selection.

Science:

No direct scientific components but involves logic and word-based gameplay.

Features:

Category selection, limited attempts, word guessing, hint system, graphical interface with buttons and images.

6) References :

Tkinter - Wikipedia
Wikipedia Summary Generator using Python Tkinter - GeeksforGeeks
GitHub - goldsmith/Wikipedia: A Pythonic wrapper for the Wikipedia API
Wikipedia Module in Python - Javatpoint