SOURCE CODE

using System;

using System.Collections;

using System.Collections.Generic;

using System.Linq;

using System.Linq.Expressions;

using System.Security.Policy;

using System.Windows.Forms;

namespace HangmanGame

{

/\*Lungile Shongwe

1 Jnauary 2023

Purpose: The purpose for this project is to design a hangman game.

\*/

//The for this class is to combine data variables and functions into one unit.

public partial class HangmanForm : Form

{

//Creating an Array

private readonly string[] wordList = { "addition", "substraction", "multiplication", "division", "remainder" ,"equal", "mathematics","algorithm","formula","plus"}; //adding wors to the array .

private string currentWord;

private ArrayList correctGuesses; //Array list being created for the list of correct guesses being made .

private ArrayList incorrectGuesses;//array list being created for the list of incorrect guesses being made.

private SortedList<string, int> scoreboard; // Sorted list being created to store the scores .

private int maxWrongGuesses = 10; // Maximum number of guesses being made .

private int wrongGuessCount; // Variable used to count the number of wrong guesses .

public HangmanForm()

{

InitializeComponent();

InitializeGame();

incorrectLabel.Text = ""; // masking

}

private void InitializeGame()

{

currentWord = SelectRandomWord();

correctGuesses = new ArrayList(); //Arraylist for the correctGuesses being created

incorrectGuesses = new ArrayList(); //Arraylist for the incorrectGuessess being created

scoreboard = new SortedList<string, int>(); //sorted list being created for the scoreboard

wrongGuessCount = 0;

DisplayWord();

ScoreBoardDisplayed();

}

//Selecting random word

private string SelectRandomWord() //function for the random word that will be selected

{

Random random = new Random();

return wordList[random.Next(wordList.Length)];

}

private void DisplayWord() //Input by user being compared.

{

wordLabel.Text = "";

foreach (char letter in currentWord)

{

if (correctGuesses.Contains(letter)) //If the current guessed letter is correct, letter is displayed

{

wordLabel.Text += letter + " ";

}

else

{

wordLabel.Text += "\_ ";

}

}

}

private void ScoreBoardDisplayed() //scoreboard being displayed

{

scoreboardListBox.Items.Clear();

foreach (var score in scoreboard)

{

scoreboardListBox.Items.Add($"{score.Key}: {score.Value} guesses");

}

}

private void CheckGuess(char guess) //checking the guessed character being inputed

{

if (currentWord.Contains(guess)) //checking to see if the correct letter has been guessed and it will be added to the word .

{

correctGuesses.Add(guess); //adding letter

}

else //checking to see if the incorrect letter has been guessed

{

incorrectGuesses.Add(guess); //adding letter

wrongGuessCount++;

}

DisplayWord();

DisplayIncorrectGuesses();

if (wrongGuessCount >= maxWrongGuesses) // Player makes more than 10 guesses game will end

{

EndGame(false);

}

else if (!wordLabel.Text.Contains("\_"))

{

EndGame(true);

}

}

private void DisplayIncorrectGuesses() // setting the incorrect letters aside.

{

incorrectLabel.Text = string.Join(", ", incorrectGuesses.Cast<char>()); // a ',' being added to separate the letters

}

private void EndGame(bool isWinner) //Determining current status for user

{

try

{

if (isWinner)

{

MessageBox.Show("WOW,GREAT! You guessed the right word."); // Correct word is guessed

scoreboard.Add(currentWord, correctGuesses.Count);

ScoreBoardDisplayed();

}

else

{

MessageBox.Show($"OOPs, you ran out of guesses. The word was: {currentWord}"); // incorrect word being guessed

}

// ScoreBoardDisplayed();

// InitializeGame();

}

catch (Exception ex)

{

MessageBox.Show($"Something went wrong: {ex.Message}", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void HangmanForm\_KeyPress(object sender, KeyPressEventArgs e) //keypress being used

{

if (char.IsLetter(e.KeyChar))

{

char guess = char.ToLower(e.KeyChar); // letters are being displayed in lower case

if (!correctGuesses.Contains(guess) && !incorrectGuesses.Contains(guess))

{

CheckGuess(guess);

}

}

}

private void RestartGame()

{

this.Hide(); //Hide the current HangmanForm form.

HangmanForm restart = new HangmanForm(); //Creates a new instance of the HangmanForm.

restart.ShowDialog(); //Displays the new instance of the HangmanForm.

this.Close(); //Closes the current HangmanForm form once the new one is displayed.

}

private void button3\_Click(object sender, EventArgs e) //Button for start

{

InitializeGame();

RestartGame();

}

private void button4\_Click(object sender, EventArgs e) //Button for finish

{

Application.Exit();

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

}

private void scoreboardListBox\_SelectedIndexChanged(object sender, EventArgs e)

{

ScoreBoardDisplayed();

}

}

}