**Number Guessing Game**

The Number Guessing Game is a simple game in which the player must guess a randomly generated number between 1 and 100 within a limited number of attempts. The game is played in the command line interface and uses the Pygame and pyttsx3 libraries to play audio files and convert text to speech, respectively.

**How it was planned and executed:**

* First of all created a outline of the program, how the flow of code should be, this is my first experience with python, so gone through google searches and done many research about the flow of code and how to run it error free. Then I had an confusion in installing the compiler, then my friend suggested me to go with Pycharm.
* First, I created a program with a simple layout, and it ran sucessfully, but I had some flaws in it, for example when I enter other than 1 to 100, it gets out of the program, by throwing an error, then I found a solution with try exception statement and this time when user enters wrong value, it asks them to enter a valid number.
* Next problem I faced is quit option, when I enter quit, it asks me to play again, but actually it should end the game, so this has taken me lot of time to solve this problem, but finally I found the solution for it.
* Next I want to improve the user experience, so I thought of playing some audios, that will make user to cheak their answer is correct or not. After googling came to know that, pygame helps to insert audio in the program that helps to create response with the audio.
* Then I thought of making the audio like human talking to you, so I tried text to speech and this option has taken my program to next level.
* Then I found the welcome audio and end audio too robotic audio, So I need a audio with a good human voice, So I used Voicemaker software for that start and end audio files. This worked awesome.
* Finally, added a new part, by asking name from the user and greeting them with their name, this helps the user to connect with the game.
* I will upload a video by trying all the possiblities in the project file for your referece

**Getting Started**

To get started, download, or clone the repository to your local machine. Make sure that you have Python 3 installed along with the Pygame and pyttsx3 libraries. You can install Pygame and pyttsx3 using pip with the following commands:

**pip install pygame**

**pip install pyttsx3**

**Or else,** go to pygame and install the packages from the library

**Audio file:**

There will be 3 files, which improves the quality of the game, by explaning the player from start to end with a pleasent greeting.

**Algorithm:**

1. Initialize Pygame mixer module and create a Pyttsx3 engine object with the 'sapi5' TTS API.
2. Set the text-to-speech engine's voice and rate properties using the Windows Registry voice ID and a rate of 125 words per minute.
3. Define the filenames of three audio files to be used in the program.
4. Define the play\_audio function that loads an audio file, plays the audio, and waits until the audio has finished playing.
5. Define the speak function that uses the pyttsx3 engine to convert text to speech and speaks the text.
6. Define the generate\_random\_number function that generates and returns a random integer between 1 and 100, inclusive.
7. Define the get\_user\_name function that prompts the user to enter their name and returns the input as a string.
8. Define the get\_user\_guess function that prompts the user to enter a guess between 1 and 100 or quit, and validates the input.
9. Define the display\_message function that displays a message based on the situation and the number of tries in the game.
10. Define the update\_high\_scores function that updates the list of high scores with the latest score and returns the top 5 scores.
11. Define the display\_high\_scores function that prints the high scores with their respective index.
12. Define the main function that runs the number guessing game. It initializes the high scores list, gets the user's name, and starts the game loop. If the user guesses the number correctly, the function updates the high scores and displays them. If the user quits the game, the function ends the game and thanks the player.

Within the main function:

1. Play welcome audio files and get the user's name.
2. Start the game loop:
3. Generate a random number between 1 and 100 and set Number\_of\_tries to 0 and is\_correct to False.
4. Start the timer.
5. While is\_correct is False:

* Get the user's guess using get\_user\_guess function.
* If the user quits the game, break out of the loop and set the situation to "quit".
* If the user guesses the number correctly, set is\_correct to True and break out of the loop.
* If the user's guess is incorrect, increment Number\_of\_tries by 1 and continue the loop.

IV. End the timer and calculate the elapsed time.

V. If the user guessed the number correctly:

* Determine the situation and display the appropriate message using the display\_message function.
* Display the elapsed time.
* If the situation is not "quit", update the high scores using the update\_high\_scores function and display the high scores using the display\_high\_scores function.

VI. If the user quit the game, display a message thanking the player and end the game.