

**PROJECT TITLE**

**MOVIE GUESSING GAME**

**CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.No** | **Topic** | **Page Number** |
| **1.** | **Problem Statement** | **3** |
| **2.** | **Problem Analysis** | **3** |
| **3.** | **Problem Design** | **4-6** |
| **4.** | **Programming Requirements** | **7-8** |
| **5.** | **Data/Input Output Description** | **9** |
| **6.** | **Algorithmic approach/Algorithm/DFD/ER diagram/Program Steps** | **10-11** |
| **7.** | **Implementation and Testing** | **12-13** |
| **8.** | **Output (Screenshots)** | **14-15** |

**Project Description**

**Problem Statement:** To make a **‘Movie Guessing Game’** Python and inculcating data visualization for its analysis.

**Problem Analysis:** A Movie Guessing game has the following to keep in consideration-

* Keeping a choice between a Single player mode and Multiplayer mode.
* Including Easy, Medium and Hard levels in the game.
* Presenting a hint for the movie and displaying the vowels while hiding the consonants.
* The most important part is displaying the wrong and right guesses in the game in order to provide a lead to the player.
* The game should also check if same letters are entered repeatedly and also that no vowels are allowed to enter by the player.
* For the Movie Analysis part displaying components like displaying the accuracy percentage and the letter frequencies along with the frequencies of vowels and consonants using bar graphs, histograms and pie charts.

**Problem Design:** This game mainly focuses on developing the predicting ability of an individual by connecting fun and knowledge. There is variable data required according to the player demands-

* For the Single Player Mode:-
* Using File Handling preparing a text file and accessing it in the programming. This file contains a premade list of movies and their respective hints.
* The movies in the file are arranged according to the level of difficulty in guessing in order to segregate them as Easy, Medium and Hard.
* For the Multiplayer Mode:-
* Every time this mode is chosen by the players, then one player is supposed to enter the Movie to be guessed and the hint and the other player just guesses it.
* In this mode the data changes every time according to the user’s wish as there is no premade list of movies.
* Irrespective of which mode is chosen by the users, there are some conditions that are followed in the game:
* A letter cannot be entered by the user more than once. In case the user inputs a choice more than once, an error message is displayed.
* Since the vowels will already be displayed while guessing the movie, the user does not need to enter vowels as their choice. In case a vowel is input by the user, an error message is displayed.
* When the letter guessed by the user is a right guess then that letter is appended to the string entered and the blank space ‘-‘ is occupied by that letter.
* If the letter guessed by the user is a wrong guess than a message is displayed and the color of the string BOLLYWOOD changes from green to red (one letter at a time) while that letter is added to the list of wrong choices.
* For the Data Visualization component there are three things taken into consideration for the content analysis:
* A pie chart shows the percentage of right and wrong guesses thereby displaying the accuracy of the player in guessing the movie.
* A bar graph displays the frequency of vowels and consonants. Often these bar graph help the player to get an understanding in making the right guesses by predicting the probabilities.
* A histogram displays the letter frequencies of the movie thereby helping the player to get an insight into the predictions made by them.

**Programming Requirements**

* **Functional Requirements:-**
* The program is constructed in **Python 3** and uses various libraries in order to execute data visualization and helps in building the data frames. The libraries used for this purpose are:
* **Matplotlib**
* **Numpy**
* **Pandas**
* In the game clear\_output method of the **IPython library**, helps in clearing a portion of the output displayed and the **time** package helps in adding a time limit in hiding the text. This function is used in the multiplayer mode where the movie and hint gets hidden after a short period so that the other player can start guessing it. The libraries and packages used are:
* **From IPython.display import clear\_output**
* **Import time**
* For the single player mode there is a premade list of movies saved in a text file. When the player chooses this mode then based on the difficulty level chosen the movies are selected at random. For this purpose the packages and libraries used are:
* **Import random**
* **From random import randint**
* In order to get a count of right and wrong guesses along with the frequency calculation the counter method is imported from collections. The text color changes due to importing the libraries to do so. The libraries imported for this purpose are:
* **From colorama import fore**
* **From collections import Counter**

**Data Input/Output Description**

The program uses the concepts of OOP and control structures in order to implement the game and solve the problems related to it.

* When the game starts running then a welcome message is shown on the output screen.
* Then the player is asked to choose between Single Player Mode and Multiplayer Mode.
* If the player chooses Single Player Mode than the player is asked to choose the difficulty level that is, Easy, Medium and Hard.
* After choosing your preferred level, the guessing game begins. In the game the user is asked to enter a choice till the number of chances are not finished that is till all the letters of the Bollywood displayed on the screen changes to red, the player inputs the choice.
* If the input choice is right than the letter gets added to the blank spaces (here denoted by “-“). If the entered guess is wrong the letters in “BOLLYWOOD” changes color (one letter at a time) from green to red and the wrong choice can be seen by the player.
* If the player chooses Multiplayer mode then it becomes a game for more than one player wherein one player inputs the movie to be guessed and its hint. Then both of these are hidden after a short while and the game begins for the second player.
* Finally, the graphs for accuracy and frequencies are displayed in order to give player an insight of how to predict the choices and displaying the performance of the player.

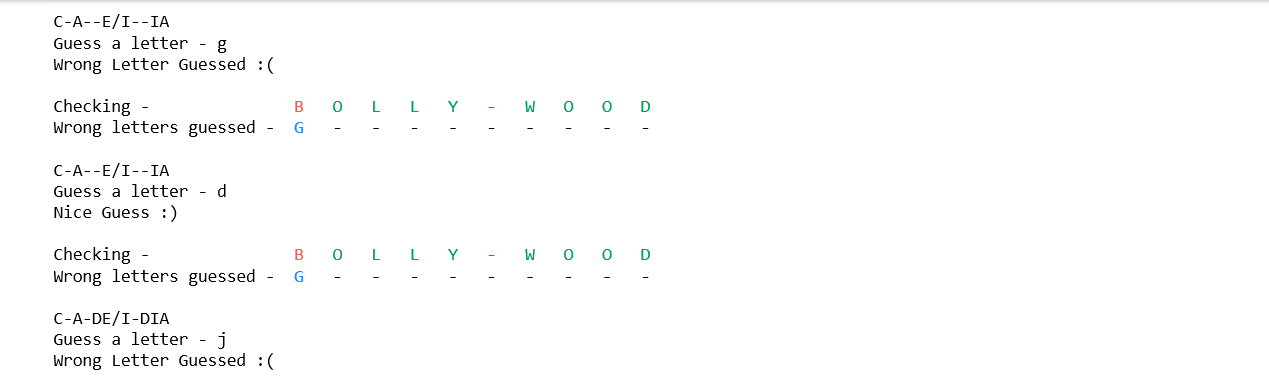
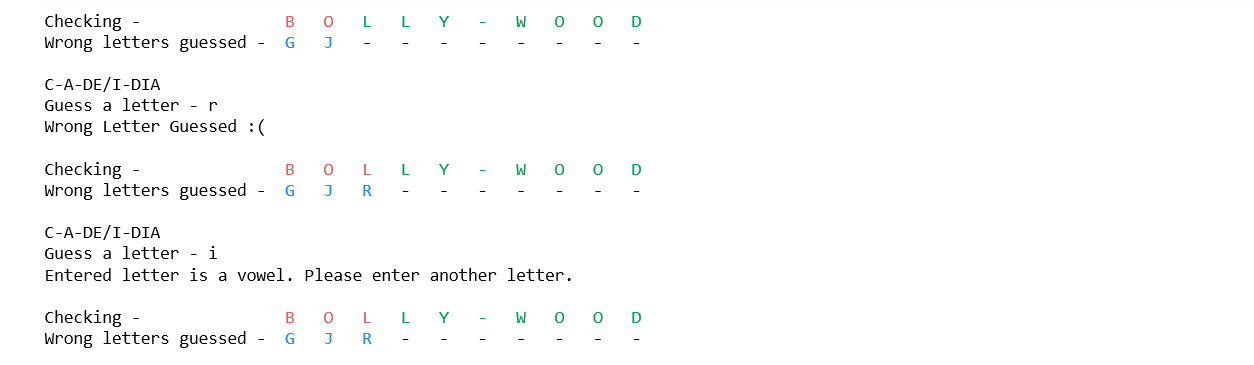
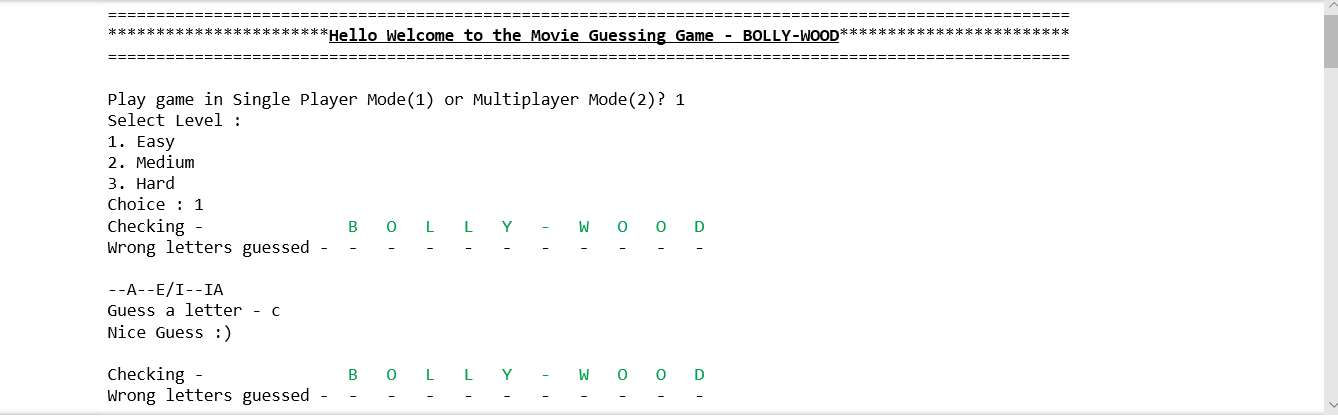
**Program Steps**

* The first step in constructing the game is making a class **Movie** that contains functions for entering the movie, entering the hint, displaying only vowels, replacing the space by ‘/’ , replacing the consonants by ‘-‘, so that guessing structure can be formed.
* Then the next step is updating and appending the correct choices in the correct blank spaces (here denoted by ‘-‘) in the given movie to be guessed.
* The next step is structuring the functionality of the game. For this, it’s required that the guesses made by the player are constantly checked and the constraints are kept in mind that is, no repetition of letters occur, no vowel is allowed to enter by the player and in case the player violates these rules than an error message is displayed. This requires making another class say **Bollywood** and defining various functions to carry out these operations.
* In the Bollywood class, also it is made sure to indicate the wrong and right choices accompanied by indicating the number of chances left by using colors.
* One of the most important parts of making the game is hiding the movie and hint after a given period of time. This is done in the clearOutput() method wherein time and clear\_output method is imported in order to carry out the operation.
* Then comes making the Single Player mode. Here, File Handling plays a crucial role. A text is created where a list of movies and hints are written and then read into the program in the selectMovie() method of the program. Then the movies are segregated using if else statements into Easy, Medium and Hard.
* The movies changes randomly due to the use of randint from random.
* Finally, comes the data visualization part wherein 3 methods are created:
* **vizShowGuessPie method is created that displays a pie chart of right and wrong guesses of the player, thereby refelecting the performance.**
* **vcFrequencyGraph method counts the number of vowels and consonants in the movie to be guessed and displays a bar graph of the same.**
* **letterFreqGraph method calculates the letter frequencies in the movie. Here the movie is first converted from a string into a list where the spaces are removed to avoid inconsistency in the graph and then copied to a dictionary in order to calculate the respective frequencies. Then finally a histogram is plotted with the help of zip(\*most\_common() ) method that unpacks the integrated dictionary and separates each letter frequency, and the most common() method here ensures that unique values are plotted and letter with the same frequencies are plotted adjacently. Basically, this ensures a sorted histogram.**
* Finally through the control structures that is the if else commands, the conditions for choice are defined that is ‘1’ for Single Player Mode, ‘2’ for Multiplayer mode and if any other number is input then an error message is printed. At last all the methods are called. The checkLose() method returns true if the guessed movie does not match the given movie and a message is printed for the same, else its printed that “You Guessed the right movie”.

**Implementation and Testing**

|  |  |  |
| --- | --- | --- |
| S.No | Cases | Testing(Screenshots) |
| 1 | Ensuring that either 1 one or 2 is entered for mode selection. Or else error message is displayed. | **Invalid case**  **Valid case** |
| 2 | Ensuring that no vowel is entered by the user by displaying an error message. |  |
| 3 | Ensuring that no repetition occurs. |  |
| 4 | Hiding output in multiplayer mode. | **First a message is displayed**    **After sometime the output is hidden** |
| 5 | Displaying the wrong guesses and adding the right guesses to the incomplete movie. | **Wrong guesses**    **Right guesses** |

**OUTPUT**

********

