ACKNOWLEDGEMENT

I would like to thank my **Principal** and the management for giving me the opportunity to do this project.

I would like to thank my Computer Teacher **Mrs. Pravitha** for constantly guiding and supporting me to do this project.

I would also like to thank my **Parents and Teammates** for their support and motivation.

We as a group consisting of Arnav, Akshya, Timon, Immanuel, Aditya and Adit of grade 10 are happy to do this project which has motivated us to be creative and innovative.

Index

About Python

Python is a high-level, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation. Python is dynamically-typed. It supports multiple programming paradigms, including structured, object oriented and functional programming.

Python is a popular programming language. it was created by Guido Van Rossum and released in 1991.

Uses of Python

- Web development
- Software development
- Gaming
- Data Visualization

Python Features: -

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems.

- **Scalable** − Python provides a better structure and support for large programs than shell scripting.
- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- Python is a Beginner's Language − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

System Specifications

- Windows or Mac Operating System
- x86 64-bit CPU (Intel / AMD / x64 architecture)
- 4 GB RAM.
- 5 GB free disk space.
- SVGA or higher-resolution monitors (XGA recommended)
- Mouse or other pointing device

Project Specification

Project Name: “Game Collection”

Description:

This project, named "Game Collection", is a collection of fun, engaging games developed in Python. It is designed for users to select and play games from a list of choices. This document aims to provide a comprehensive understanding of the project's structure, the games included, the code involved, and various Python functionalities used in building the project.

The default games included are:

1. Mastermind: A classic game where the player tries to guess a random 4-digit number. Feedback is given on each guess to assist the player in making subsequent guesses.
2. Jumbled Word Game: Players are given a jumbled word, and they must unscramble it. If unsuccessful, then the chance is passed on to the next player.
3. Trivia Quiz Game: Players are given a multiple-choice question pertaining to different categories such as Sports, Music, Arts, Technology, “General Knowledge”. These questions are taken from a Trivia Quiz API and are unique for each new game.

How to use?

After successfully installing the project, run the following command in your terminal or command prompt:

```
python main.py
```

Upon launching the program, you'll be presented with a main menu showing a list of games available:

1. Mastermind
2. Jumbled Word Game
3. Trivia Quiz
4. Quit

To select a game, simply input the corresponding number.

Mastermind:

Objective: Guess the correct sequence of colors.

1. You have a certain number of attempts to guess the correct sequence.
2. After each guess, you'll receive feedback in the form of colored pegs

Jumbled Word Game:

Objective: Rearrange the scrambled letters to form a valid word.

1. A jumbled word will be shown on the screen.
2. Input your guess and check if it's correct.

Trivia Quiz:

Objective: Guess the word with a limited number of attempts.

1. You'll be given a multiple-choice question with one answer being correct.
2. With each correct answer, you will be given another question.
3. A wrong answer will quit you from the game after showing your score.

Exiting a Specific Game:

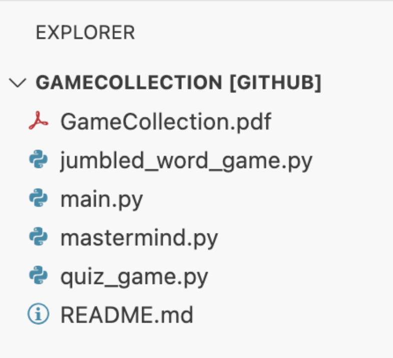
For all games, you can quit and return to the main menu. If a game ends (win or lose), you'll be presented with an option to play again or return to the main menu.

Quitting the Game:

To quit the entire Game Collection, select “**4. Quit**” from the main menu or simply close the terminal or command prompt.

The code for the project can be accessed at GitHub repository:
https://github.com/adityogi/GameCollection

Folder Structure:



EXPLORER

GAMECOLLECTION [GITHUB]
- GameCollection.pdf
- jumbled_word_game.py
- main.py
- mastermind.py
- quiz_game.py
- README.md

Prebuilt Libraries/Packages used:

1. **random:**

   This module implements pseudo-random number generators for various distributions. Functions like randint and choice are used from this module.

   a. randint(a, b): Return a random integer between a and b (inclusive).

   b. choice(sequence): Return a random element from the sequence.

2. **subprocess:**

   Allows spawning new processes, connecting to their input/output/error pipes, and obtaining their return codes.

   a. run(args): Used to run the shell commands. In this project, it is used to execute Python game files.

3. **requests:**

   The requests module allows you to send HTTP requests using Python.

   a. requests.get(URL) allows you to invoke a HTTP GET Request for any URL

   b. The HTTP request returns a Response Object with all the response data (content, encoding, status).

Summary

This Game Collection project is a delightful ensemble of various games designed for pure entertainment. Written in Python, it leverages various libraries and constructs. The code is modular, with each game having its Python file, making it easier to understand and manage.

File 1: main.py

Code:

```
import subprocess


def main():
    # Start an infinite loop to keep displaying the menu until the user decides to quit.
    while True:
        # Printing the options for the user to select the game.
        print("Choose a game to play:")
        print("1. Mastermind")
        print("2. Jumbled Word Game")
        print("3. Trivia Quiz")
        print("4. Quit")

        # Taking the user's choice.
        choice = input("Enter your choice (1/2/3/4): ")

        # Based on the user's choice, run the corresponding game using subprocess.
        # subprocess.run() is used to run shell commands, in this case, to execute the
Python game files.

        if choice == "1":
            # Run the mastermind game.
            subprocess.run(["python", "mastermind.py"])
        elif choice == "2":
            # Run the jumbled word game.
            subprocess.run(["python", "jumbled_word_game.py"])
        elif choice == "3":
            # Run the quiz game.
            subprocess.run(["python", "quiz_game.py"])
        elif choice == "4":
            # Exit the loop, thereby ending the program.
            print("Thanks for playing!")
            break
        else:
            # If the user enters a choice not listed in the menu, inform them and display
the menu again.
            print("Invalid choice. Please choose again.")


# The code within this conditional block will only execute if main.py is run directly
(and not imported elsewhere).
if __name__ == "__main__":
    main()
```

File 2: mastermind.py

Code:

```
import random

# the .randrange() function generates
# a random number within the specified range.
NUM_TRIES = 10
NUM_DIGITS = 4
def welcome():
    print("\n")
    print("*"*15, "Welcome to MasterMind", "*"*15)
    print(f"""  *** RULES ***
  Computer has thought of a {NUM_DIGITS} digit number
  You have {NUM_TRIES} tries to guess it
   => Press q or x to exit
   => Press n to reset & start a new game
   """)
    print("*"*53)

def get_user_input():
    while True:
        user_input = input(f"Guess the {NUM_DIGITS} digit number: ")
        if user_input in ['q', 'Q', 'x', 'X']:
            print("Sorry to see you go")
            exit(0)
        # elif user_input in ['n', 'N']:
        #   new_game()
        elif validate(user_input):
            return user_input

def validate(user_input):
    try:
        user_guess = int(user_input)
        if user_guess < pow(10, NUM_DIGITS - 1) or user_guess > pow(10, NUM_DIGITS):
            print(f"Provide {NUM_DIGITS} digit number only")
            return False
        else:
            return True
    except Exception as e:
        print("Try again, input is invalid:", e)

def new_game():
    num = random.randrange(pow(10, NUM_DIGITS - 1), pow(10, NUM_DIGITS))
    # print("random number", num)
    play_game(num)

def play_game(num):
    tries = 0
    while True:
        user_input = get_user_input()
        tries += 1
        # print(user_input)
        correct_digits = compare(user_input, num)
        if  correct_digits == NUM_DIGITS:
            print(f"Guessed it right. You took just {tries} attempts")
            break
        elif tries >= NUM_TRIES and correct_digits == 0:
            print(f'Oh, too many attempts. How about starting afresh?')
            print('Computer number was', num)
            new_game()

def compare(user_guess, computer_num):
    if (user_guess == computer_num):
        return len(str(user_guess))

    str_user_guess = str(user_guess)
    str_computer_num = str(computer_num)
    index = 0
    correct = 0
    print_matching = list()
    digit_found = list()
    while index < len(str_user_guess):
        if str_user_guess[index] != str_computer_num[index]:
            print_matching.append('_')
        else:
            print_matching.append(str_user_guess[index])
            correct += 1
        if str_user_guess[index] in str_computer_num:
            digit_found.append(str_user_guess[index])
        index += 1
    if correct > 0:
        print("Your number matches partially:", ''.join(print_matching))
    else:
        print("None of the digits match")
    if len(digit_found) > 0:
        print("These digits are common to both numbers:", ','.join(digit_found))
    return correct

def main():
    welcome()
    new_game()

if __name__ == "__main__":
    main()
```

File 3: jumbled_word_game.py

Code:

```python
# Python program for jumbled words game.

# import random module
import random

# Global scope
players = list()
score = dict()


# function for choosing random word.
def choose_a_word():
    # list of word
    words = ['rainbow', 'computer', 'science', 'programming',
            'mathematics', 'player', 'condition', 'reverse',
            'water', 'board', 'account', 'administration', 'affect', 'agency', 'beyond']

    # choice() method randomly choose
    # any word from the list.
    pick = random.choice(words)
    return pick


# Function for shuffling the
# characters of the chosen word.
def jumble_it(word):
    # sample() method shuffling the characters of the word
    random_word = random.sample(word, len(word))

    # join() method join the elements
    # of the iterator(e.g. list) with particular character .
    jumbled = ''.join(random_word)
    return jumbled

# Function for declaring winner
def check_win():
    winner = ''
    winner_score = 0
    # # hits = [ v for v in score.values() ]
    # # biggest_hit = hits.sort(reverse=True)[0]
    # winners = list()
    for (player, strike) in score.items():
        if strike < winner_score:
            winner_score = strike
            winner = player
        # if strike == biggest_hit:
        #   winners.append(player)
    if winner == '':
        print('Nobody won')
    else:
        print(f'{winner} won with {winner_score} hits')
    # print(','.join(winners), 'got the highest score', biggest_hit)



# Function for showing final score.
def thank_you_all():
    # print("Thanks to ", ",".join(players))
    winning_score = 0
    per_score = dict()
    for p in players:
        print(f"{p}'s score is {score[p]}")
        if not per_score.get(score[p]):
            per_score[score[p]] = [p]
        else:
            per_score[score[p]].append(p)
        if score[p] > winning_score:
            winning_score = score[p]

    print(f'Winning score is {winning_score}')
    print('And the winner(s) are', ','.join(per_score[winning_score]))
    print('Thanks for playing !!!', ",".join(players).upper())

def ask(player_name, turn, picked_word):
    # print(f'"{player_name}", it is your Turn.')
    ans = input(f" * [Word:{turn}] {player_name}, what do you think, it is? =>")
    return (ans == picked_word)

# def get_player_turn(player_list, turn):
#     return (len(player_list) % turn)

def continue_playing():
    user_response = input("""Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to
quit ...) <===
""")
    if user_response and user_response.upper() == 'X':
        return False
    else:
        return True

def play_turn(turn, picked_word):
        num_players = len(players)
        player_position = turn % num_players
        # player_position = original_chance

        player_order = players[player_position:] + players[:player_position]
        got_it = False
        for player in player_order:
            if (ask(player, turn, picked_word)):
```

```python
                print(f'You got it right, {player}')
                score[player] += 1
                got_it = True
                break

        if not got_it:
            print("Better luck next time...correct word is :", picked_word)

def play_game():
    # variable for counting turn
    turn = 0
    while continue_playing():
        picked_word = choose_a_word()
        scrambled = jumble_it(picked_word)
        print("jumbled word is :", scrambled)
        # original_chance = get_player_turn(players, turn)
        play_turn(turn, picked_word)
        turn += 1
    print(f'Played {turn} scrambled words')

# Function for playing the game.
def setup_game():
    while True:
        user_txt = input("Enter Player Name OR xx to finish adding players:")
        if user_txt.lower() == 'xx':
            break
        elif user_txt != '':
            if user_txt not in players:
                players.append(user_txt)

    num_players = len(players)
    if num_players > 0:
        print('Players:', players)
        for p in players:
            score[p] = 0
    else:
        print("Sorry, can't play without players")
        return

    # Play the game, now that we have players
    play_game()

    # Exit by showing all the scores and winner(s)
    thank_you_all()

# Driver code
if __name__ == '__main__':
    setup_game()
```

File 4: quiz_game.py

Code:

```python
import requests
import random

#  Defining function for game playing
def play_game_kb(username, kb):
    print("Hello,", username, "welcome to the QUIZ game")
    print("All the Best for the Game :>")
    score = 0
    random.shuffle(kb)
    exit = False
    for qset in kb:
        current_question = qset['question']
        correct_answer = qset['correctAnswer']
        current_question_options = qset['incorrectAnswers']
        if correct_answer not in current_question_options:
            current_question_options.append(correct_answer)
        random.shuffle(current_question_options)
        print("Question:", current_question)
        user_options = list()
        for index, each_options in enumerate(current_question_options):
            print(index+1, ") ", each_options, sep='')
            user_options.append(str(index+1))
        exit_option = len(current_question_options)+1
        print(exit_option, ") ", 'Exit', sep='')
        user_options.append(str(exit_option))

        user_answer_index = 0
        option_list = ','.join(user_options)

        while str(user_answer_index) not in user_options:
            try:
                user_answer_index = int(input(f"Please enter your choice({option_list}):
"))
            except Exception as e:
                print(f"Invalid input => [{e}]")
        if user_answer_index == exit_option:
            print("Bye\n")
            exit = True
            break
        user_answer = current_question_options[user_answer_index-1]
        if user_answer == correct_answer:
            print("correct answer\n")
            score = score + 100
        else:
            print("Wrong Answer!!! \n => Correct answer is", correct_answer)
            break
    print("Your final score is", score)
    return score, exit
```

```python
# Defining function for viewing the score
def view_scores(names_and_scores):
    for name, score in names_and_scores.items():
        print(name, "has scored", score)

# Defining the function for start of the score
def quiz_kb():
    names_and_scores = dict()
    while True:
        print("\nWelcome to the quiz game")
        print("1) Play\n2) View Scores\n3) Exit")
        choice = int(input("Please enter your choice: "))
        if choice == 1:
            username = ''
            while username.strip() == "":
                username = input("Please enter your name: ")
            knowledge_base = setup_kb()
            score, exit = play_game_kb(username, knowledge_base)
            names_and_scores[username] = score
            if exit:
                break

        elif choice == 2:
            view_scores(names_and_scores)
        elif choice == 3:
            break
        else:
            print("Your choice is not correct")


# get quiz questions
def setup_kb():
    knowledge_base = list()
    qset = {
        "category": "Programming",
        "id": "",
        "correctAnswer": "Guido Van",
        "incorrectAnswers": [
        "Dennis Ritchie",
        "Alan Frank",
        "Albert"
        ],
        "question": "Who is the developer of Python Language?",
        "tags": [
        "technical",
        "programming",
        "python"
        ],
        "type": "Multiple Choice",
        "difficulty": "medium",
        "regions": [],
        "isNiche": False
    }
    response = requests.get('https://the-trivia-api.com/api/questions/')
    if response.status_code == 200:
        knowledge_base = response.json()
    else:
        knowledge_base.append(qset)
    return knowledge_base

#  Program execution starts from here
if __name__ == '__main__':
    quiz_kb()
```

Output for the Program & Playing the Game:

main.py

```
python3 main.py
Choose a game to play:
1. Mastermind
2. Jumbled Word Game
3. Trivia Quiz
4. Quit
Enter your choice (1/2/3/4): 3

Welcome to the quiz game
1) Play
2) View Scores
3) Exit
Please enter your choice: 1
Please enter your name: Adi
Hello, Adi welcome to the QUIZ game
All the Best for the Game :>
Question: How Many Movements Traditionally Make Up A Concerto?
1 )  4
2 )  2
3 )  3
4 )  5
5 )  Exit
Please enter your choice(1,2,3,4,5): 4
Wrong Answer!!!
 => Correct answer is 3
Your final score is 0

Welcome to the quiz game
1) Play
2) View Scores
3) Exit
Please enter your choice: 1
Please enter your name: Kris
Hello, Kris welcome to the QUIZ game
All the Best for the Game :>
Question: Which author wrote 'The Overcoat'?
1 )  Ayn Rand
2 )  Virginia Woolf
3 )  Nikolai Gogol
4 )  Alexander Pushkin
5 )  Exit
Please enter your choice(1,2,3,4,5): 2
Wrong Answer!!!
 => Correct answer is Nikolai Gogol
Your final score is 0

Welcome to the quiz game
1) Play
2) View Scores
3) Exit
Please enter your choice: 1
Please enter your name: Hari
Hello, Hari welcome to the QUIZ game
All the Best for the Game :>
Question: Which Is The Largest State In Australia
1 )  Western Australia
```

```
2 )  Tasmania
3 )  Victoria
4 )  Queensland
5 )  Exit
Please enter your choice(1,2,3,4,5): 2
Wrong Answer!!!
 => Correct answer is Western Australia
Your final score is 0

Welcome to the quiz game
1) Play
2) View Scores
3) Exit
Please enter your choice: 2
Adi has scored 0
Kris has scored 0
Hari has scored 0

Welcome to the quiz game
1) Play
2) View Scores
3) Exit
Please enter your choice: 3
Choose a game to play:
1. Mastermind
2. Jumbled Word Game
3. Trivia Quiz
4. Quit
Enter your choice (1/2/3/4): 1


*************** Welcome to MasterMind ***************
  *** RULES ***
  Computer has thought of a 4 digit number
  You have 10 tries to guess it
   => Press q or x to exit
   => Press n to reset & start a new game

*****************************************************
Guess the 4 digit number: 1234
None of the digits match
These digits are common to both numbers: 1,2
Guess the 4 digit number: 2156
Your number matches partially: 21__
These digits are common to both numbers: 2,1
Guess the 4 digit number: 2178
Your number matches partially: 217_
These digits are common to both numbers: 2,1,7
Guess the 4 digit number: 2179
Your number matches partially: 2179
These digits are common to both numbers: 2,1,7,9
Guessed it right. You took just 4 attempts
Choose a game to play:
1. Mastermind
2. Jumbled Word Game
3. Trivia Quiz
4. Quit
Enter your choice (1/2/3/4): 2
Enter Player Name OR xx to finish adding players:Adi
Enter Player Name OR xx to finish adding players:Kris
Enter Player Name OR xx to finish adding players:xx
```

```
Players: ['Adi', 'Kris']
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===

jumbled word is : mtarinoinitasd
 * [Word:0] Adi, what do you think, it is? =>administration
You got it right, Adi
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===

jumbled word is : tmhmactsiae
 * [Word:1] Kris, what do you think, it is? =>mathematics
You got it right, Kris
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===

jumbled word is : ygance
 * [Word:2] Adi, what do you think, it is? =>agency
You got it right, Adi
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===
x
Played 3 scrambled words
Adi's score is 2
Kris's score is 1
Winning score is 2
And the winner(s) are Adi
Thanks for playing !!! ADI,KRIS
Choose a game to play:
1. Mastermind
2. Jumbled Word Game
3. Trivia Quiz
4. Quit
Enter your choice (1/2/3/4): 4
Thanks for playing!
```

quiz_game.py

```
python3 quiz_game.py

Welcome to the quiz game
1) Play
2) View Scores
3) Exit
Please enter your choice: 1
Please enter your name: Adi
Hello, Adi welcome to the QUIZ game
All the Best for the Game :>
Question: In Which City Was John F Kennedy Assassinated?
1) Atlanta
2) Houston
3) Austin
4) Dallas
5) Exit
Please enter your choice(1,2,3,4,5): 4
correct answer

Question: Which of these iconic Russian buildings is located in Moscow?
1) St. Basil's Cathedral
2) Winter Palace
3) Kazan Cathedral
4) Peter and Paul Fortress
5) Exit
Please enter your choice(1,2,3,4,5): 1
correct answer

Question: What vegetable is known a zucchini in the USA?
1) Onion
2) Cucumber
3) Carrot
4) Courgette
5) Exit
Please enter your choice(1,2,3,4,5): 4
correct answer

Question: What is the name of a shape with eight sides?
1) Triangle
2) Rectangle
3) Octagon
4) Hexagon
5) Exit
Please enter your choice(1,2,3,4,5): 3
correct answer

Question: Which song begins with the lyrics: "I said a hip-hop, the hippie the
hippie / To the hip, hip hop you don't stop / Rock it to the bang bang boogie /
Say up jump the boogie to the rhythm of the boogie, the beat..."?
1) "Rapper's Delight" by Sugarhill Gang
2) "You're So Vain" by Carly Simon
3) "Gimme More" by Britney Spears
4) "Man! I Feel Like a Woman!" by Shania Twain
5) Exit
Please enter your choice(1,2,3,4,5): 5
Bye

Your final score is 400
```

mastermind.py

```
python3 mastermind.py


*************** Welcome to MasterMind ***************
  *** RULES ***
  Computer has thought of a 4 digit number
  You have 10 tries to guess it
   => Press q or x to exit
   => Press n to reset & start a new game

*****************************************************
Guess the 4 digit number: 3478
None of the digits match
Guess the 4 digit number: 1256
None of the digits match
These digits are common to both numbers: 2,6
Guess the 4 digit number: 2609
Your number matches partially: _6__
These digits are common to both numbers: 2,6,9
Guess the 4 digit number: 9622
Your number matches partially: 96_2
These digits are common to both numbers: 9,6,2,2
Guess the 4 digit number: 9662
Your number matches partially: 96_2
These digits are common to both numbers: 9,6,6,2
Guess the 4 digit number: 9692
Your number matches partially: 9692
These digits are common to both numbers: 9,6,9,2
Guessed it right. You took just 6 attempts
```

jumbled_word.py

```
python3 jumbled_word_game.py
Enter Player Name OR xx to finish adding players:Adi
Enter Player Name OR xx to finish adding players:Kris
Enter Player Name OR xx to finish adding players:Hari
Enter Player Name OR xx to finish adding players:M
Enter Player Name OR xx to finish adding players:xx
Players: ['Adi', 'Kris', 'Hari', 'M']
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===

jumbled word is : yegcan
 * [Word:0] Adi, what do you think, it is? =>agency
You got it right, Adi
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===

jumbled word is : genacy
 * [Word:1] Kris, what do you think, it is? =>agency
You got it right, Kris
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===

jumbled word is : rawte
 * [Word:2] Hari, what do you think, it is? =>wate
 * [Word:2] M, what do you think, it is? =>water
You got it right, M
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===

jumbled word is : eynodb
 * [Word:3] M, what do you think, it is? =>boneyd
 * [Word:3] Adi, what do you think, it is? =>beyond
You got it right, Adi
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===

jumbled word is : bdora
 * [Word:4] Adi, what do you think, it is? =>broad
 * [Word:4] Kris, what do you think, it is? =>board
You got it right, Kris
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===

jumbled word is : seerevr
 * [Word:5] Kris, what do you think, it is? =>reserve
 * [Word:5] Hari, what do you think, it is? =>reverse
You got it right, Hari
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===

jumbled word is : mcuporet
 * [Word:6] Hari, what do you think, it is? =>carpet
 * [Word:6] M, what do you think, it is? =>computer
You got it right, M
Ready to PLAY JUMBLED WORDS !!!      ===>  (press 'x' to quit ...) <===
x
Played 7 scrambled words
Adi's score is 2
Kris's score is 2
Hari's score is 1
M's score is 2
Winning score is 2
And the winner(s) are Adi,Kris,M
Thanks for playing !!! ADI,KRIS,HARI,M
```

Bibliography:

i. https://www.python.org/about/gettingstarted/
ii. https://pypi.org/project/random2/
iii. https://docs.python.org/3/library/subprocess.html
iv. https://pypi.org/project/subprocess.run/
v. https://pypi.org/project/requests/
vi. https://requests.readthedocs.io/en/latest/
vii. https://github.com/kishanrajput23/Awesome-Python-Projects
viii. https://the-trivia-api.com/docs/v2/