MAD LIBS GAME

A PROJECT REPORT

Submitted by

***MOHANA HARINI* S**

(2303811710422097)

in partial fulfillment for the completion of the course

CGB1121- PYTHON PROGRAMMING

in

COMPUTER SCIENCE AND ENGINEERING

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY

SAMAYAPURAM – 621 112

JUNE, 2024

K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report titled **"MAD LIBS GAME"** is the bonafide work of **MOHANA HARINI.S (2303811710422097)** who carried out the project under my supervision. Certified further, that to the best of my knowledge the work reported here in does not form part of any other project report or dissertation on the basis of which a course was conferred on an earlier occasion on this or any other candidate.

SIGNATURE
Mrs.A.Delphin Carolina Rani,
M.E.,Ph.D.,
HEAD OF THE DEPARTMENT

PROFESSOR Department

of CSE K. Ramakrishnan

College of
Technology (Autonomous)
Samayapuram – 621 112.

SIGNATURE

Mrs. P. Sudha M.E.,Ph.D
SUPERVISOR

ASSISTANT PROFESSOR

Department of ECE

K.Ramakrishnan College of
Technology (Autonomous)
Samayapuram – 621 112.

Submitted for the viva-voce examination held on …….

DECLARATION

I declare that the project report on "MAD LIBS GAME" is the result of original work done by us and best of our knowledge, similar work has not been submitted to "ANNA UNIVERSITY CHENNAI" for the requirement of Degree of BACHELOR OF ENGINEERING. This project report is submitted on the partial fulfilment of the requirement of the completion of the course CGB1121- PYTHON PROGRAMMING.

Signature

MOHANA HARINI S

Place: Samayapuram

Date:

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "K.Ramakrishnan College of Technology (Autonomous)", for providing us with the opportunity to do this project.

I glad to credit honourable chairman Dr. K. RAMAKRISHNAN, B.E., for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director Dr. S. KUPPUSAMY, MBA, Ph.D., for forwarding to our project and offering adequate duration in completing our project.

I would like to thank Dr. N. VASUDEVAN, M.Tech., Ph.D., Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to Head of the Mrs. A. DELPHIN CAROLINA RANI, M.E.,Ph.D., COMPUTER SCIENCE AND ENGINEERING for her encourage pursuing this project. providing

I express our deep expression and sincere gratitude to our project guide Mrs. P.SUDHA, M.E.,Ph.D., Department of ELECTRONIC COMMUNICATION AND ENGINEERING, for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To emerge as a leader among the top institutions in the field of technical education.

MISSION OF THE INSTITUTION

- Produce smart technocrats with empirical knowledge who can surmount the global challenges.
- Create a diverse, fully-engaged, learner-centric campus environment to provide quality education to the students.
- Maintain mutually beneficial partnerships with our alumni, industry, and Professional associations.

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society:To empower students with the required skills to solve complex technological problability of society of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM OUTCOMES (POs)
Engineering students will be able to:

1. Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences

3. Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations

4. Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. Modern tool usage: Create, select, and apply appropriate techniques, resources, and engineering and IT tools including prediction and modeling to complex engineering activities
with an understanding of the
limitations
6. The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice

7. Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development

8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

Mad Libs is a popular word game where players fill in blanks in a story template with random words, often resulting in humorous and nonsensical narratives. This project involves developing a Mad Libs game using Python, which provides an interactive and user-friendly experience.The game starts by presenting the user with a story template containing several placeholders for different parts of speech, such as nouns, verbs, adjectives, and adverbs. The user is prompted to input words corresponding to these placeholders. Once all inputs are received, the program dynamically constructs and displays the complete story by inserting the user-provided words into the template. The project demonstrates fundamental programming concepts such as input/output operations, data validation, string manipulation, and control structures. Additionally, it offers an engaging way to practice these concepts and can be further extended with additional features, such as saving stories, sharing results, or incorporating more complex templates.Overall, the Mad Libs game in Python combines creativity and programming, providing an enjoyable learning experience for beginners and seasoned developers alike.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS		
IDE VS	-	IntegratedDevelopment
Code re	-	Environment Visual Studio Code
iOS	-	Regular Expression Iphone
NLP	-	Operating System
GUI	-	Natural Language
APIs	-	Processing

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION TO PYTHON

1.1.1. Overview

Python is a widely-used, high-level programming language renowned for its readability and simplicity, making it an ideal choice for both novice and seasoned programmers. Created by Guido van Rossum and released in 1991, Python's core philosophy emphasizes code readability and straightforward syntax, allowing developers to write clear and concise code more efficiently compared to other languages like C++ or Java.

1.1.2. Programming Paradigms

Python supports various programming paradigms, including procedural, object-oriented, and functional programming. This flexibility, combined with a dynamic type system and automatic memory management, facilitates the development of a wide range of applications, from simple scripts to complex software systems.

1.1.3. Standard Library

The language's comprehensive standard library, often referred to as "batteries-included," provides built-in modules and functions for handling many programming tasks, such as file I/O, system calls, and even web services. This extensive library helps streamline the development process by offering ready-to-use solutions for common programming challenges.

1.1.4. Third-Party Libraries And Frameworks

One of Python's significant strengths is its extensive ecosystem of third-party libraries and frameworks. Popular libraries such as NumPy and Pandas enable efficient data manipulation and analysis, while frameworks like Django and Flask streamline web development. In the realm of machine learning and artificial intelligence, libraries like TensorFlow and PyTorch are widely adopted for building and deploying sophisticated models.

1.1.5. Versions Of Python

Python has undergone significant evolution since its inception, with two major versions in use today:

Python 2: Released in 2000, Python 2.x series was a major milestone and widely used for many years. However, it reached its end of life on January 1, 2020, and is no longer maintained.

Python 3: Introduced in 2008, Python 3.x series brought substantial improvements and changes to the language, such as better Unicode support, a more consistent syntax, and enhanced standard libraries. Python 3 is the recommended version for all new projects.

1.1.6. Python Tools

Python's ecosystem includes numerous tools that enhance productivity and development experience:

- IDEs and Code Editors: Popular options include PyCharm, VS Code, and Jupyter Notebook, which offer features like syntax highlighting, code completion, and debugging.
- Package Management: Tools like pip and conda facilitate the installation an management
- of Python libraries and dependencies. to create isolated environments for different projects, ensuring dependency conflicts are avoided.
- Testing Frameworks: unittest, pytest, and nose are commonly used for writing and running Virtual Environments: virtualenv and venv allow developers tests to ensure code reliability and correctness.
- Build Tools: setuptools and wheel help in packaging Python projects, making them easy distribute and install.
- Documentation Generators: Tools documentation for Python projects.
- Linters and Formatters: pylint, flake8, and black help maintain code quality and consistency by enforcing coding standards and formatting.

1.1.7. Versatility And Adoption

Python's simplicity and versatility have led to its widespread adoption in various fields, including web development, data science, artificial intelligence, automation, and scientific computing. Its active community continually contributes to a rich repository of resources, tutorials, and documentation, making it easier for developers to learn and apply Python effectively.

CHAPTER 2 PROJECT DESCRIPTION

2.1. PROJECT INTRODUCTION

The project, "MAD LIBS GAME," aims where players fill in blanks in a story with random words without knowing the context. In Python, you can create a Mad Libs game by defining a story template with placeholders for words, then prompting the user to input words for those placeholders. Finally, you fill in the blanks with the user's inputs and display the completed story.

2.2. PROJECT OBJECTIVE

The objective of creating a Mad Libs game in Python is to practice string manipulation, user input processing, and basic programming concepts such as variables, loops, and conditionals in a fun and interactive way.

2.3. PROBLEM STATEMENT

The problem statement for creating a Mad Libs game in Python involves developing a program that prompts users to input various word types (nouns, verbs, adjectives, etc.) to fill in placeholders in a story template. The program should then combine the user inputs with the story template to create a humorous or nonsensical story for entertainment purposes.

2.4. LIBRARIES USED

The following Python libraries are utilized in this project to achieve the desired

- string (for Template strings):The string library in Python provides the Template class, which can be helpful for creating templates with placeholders that can be replaced with user input.

- re (Regular Expressions):For more complex text manipulation and extraction, the re library can be used to find and replace parts of the text.

- Basic I/O (No additional libraries needed):You can use basic Python input and print functions for a simple Mad Libs game. This involves prompting the user for different types of words and inserting them into a predefined story template.

CHAPTER 3
SYSTEM ANALYSIS

3.1. EXISTING SYSTEM

You can create a Mad Libs game in Python by using input() to prompt the user for words and then inserting those words into a predefined story template.This prompts the user for a noun, verb, and adjective, then inserts them into the mad lib sentence and prints the final story. You can expand on this by creating more elaborate templates and adding more user inputs.

3.1.1. DISADVANTAGES:

1. Static Interaction: Most text-based Mad Libs games offer limited interactivity and engagement, which might not be as appealing compared to graphical or multimedia versions.
2. Input Validation: Users might enter inappropriate or irrelevant words that do not fit well within the context of the story, leading to nonsensical or less enjoyable outputs.
3. Spelling Mistakes: Typographical errors can affect the quality of the generated story.
4. Language Skills: While Mad Libs can help with understanding parts of speech, they may not effectively teach deeper language skills or grammar rules.
5. Lack of Visuals: Text-only games might not be as engaging as those with visual or audio elements.

3.2. PROPOSED SYSTEM

One way to create a Mad Libs game in Python is to prompt users for different types of words (nouns, verbs, adjectives, etc.), store these inputs in variables, and then insert them into a pre-written story template at specified locations. The final output will be a humorous or nonsensical story created by the user's word choices.Another approach is to create a dictionary or list of placeholders in a story template and then iterate through each placeholder, prompting the user for the corresponding type of word to fill in. After collecting all the user inputs, the program can substitute them into the story template to generate the final Mad Libs story.

3.2.1. ADVANTAGES

1. Grammar Practice: It helps users understand parts of speech (nouns, verbs, adjectives, adverbs) by requiring them to identify and use them correctly.
2. Vocabulary Building: Users expand their vocabulary as they think of different words to fit the story.
3. Creativity: Encourages creative thinking as users come up with interesting and humorous combinations.
4. Simplicity: Writing a basic Mad Libs game in Python is straightforward and requires only fundamental programming concepts such as input/output, strings, and functions.
5. Beginner-Friendly: A Mad Libs game is an excellent project for beginners to practice Python programming, reinforcing basic concepts in a fun way.
6. Adaptable: The game can be adapted for different age groups, educational purposes, or themes (e.g., holiday-themed Mad Libs).
7. Template Variability: Multiple templates can be easily incorporated, making the game more dynamic and replayable.

CHAPTER 4
SYSTEM DESIGN & MODULES

4.1. BLOCK DIAGRAM

4.2. MODULE DESCRIPTION

4.2.1. INPUT TEXT MODULE

1. Start Game: Initiates the game and displays a welcome message.
2. Display Instructions: Provides the user with instructions on how to play the game.
3. Prompt for Words: Asks the user for various parts of speech needed for the story template.
4. Insert Words in Story: Takes the user input and replaces placeholders in the story template.
5. Display Final Story: Shows the completed story with the user's words inserted, creating a fun and engaging narrative.
6. End Game: Thanks the user for playing and ends the game session.

- Takes a list of prompts (parts of speech) and asks the user to input words corresponding to each prompt. Returns a dictionary with the prompts as keys and the user inputs as values.
- The script includes an example usage of the get_ user_ inputs function. It defines a list of prompts and calls the function to collect user inputs. After collecting the inputs, it prints the collected inputs for verification.
- The prompts list contains the parts of speech needed for the story template. This list should match the placeholders in the story template.

4.2.2. OUTPUT TEXT MODULE

This module provides the final output text, where Mad Libs Game have been replaced with their textual descriptions. The processed text can be returned to the user or passed to other applications for further processing.

- Takes a story template with placeholders and a dictionary of user inputs.
- Replaces each placeholder in the template with the corresponding user input.
- Returns the final story.
- The script includes an example usage of the create_ story function.
- It defines a story template with placeholders and a dictionary of user inputs.
- The placeholders in the story template are denoted by angle brackets

CHAPTER 5 CONCLUSION & FUTURE ENHANCEMENT

5.1. CONCLUSION

Developing a Mad Libs game in Python provides an engaging way to practice language skills and programming concepts. The modular approach outlined in this document ensures that the game is easy to understand, maintain, and extend. Each module has a specific responsibility, from loading templates and collecting user inputs to generating and displaying the final story. This structure not only facilitates the development process but also enhances the game's flexibility and replayability. Potential enhancements could include adding more complex templates, improving input validation, and incorporating a graphical user interface (GUI) to make the game more interactive and visually appealing. Overall, the Mad Libs game in Python is a fun and educational project that highlights the power and simplicity of Python programming.

5.2. FUTURE ENHANCEMENT

- GUI Implementation: Developing a graphical user interface for a more user-friendly experience.
- Story Templates: Expanding the number of available story templates.
- Word Suggestions: Providing word suggestions to users to aid in word selection.
- Online Play: Enabling multiplayer functionality for online play.
- Save and Share: Allowing users to save and share their completed stories.

By implementing these enhancements, the Mad Libs game in Python can evolve into a more sophisticated, user-friendly, and widely accessible application, catering to diverse audiences and maintaining long-term engagement. In the future, the Mad Libs game in Python could undergo several enhancements to elevate its user experience and functionality. One potential enhancement involves the integration of a graphical user interface (GUI) using libraries like Tkinter or PyQt, providing a more visually appealing and intuitive interface for players to interact with.

Additionally, dynamic template loading from an online repository could be implemented, ensuring a constantly updated library of templates without requiring manual updates. Advanced input validation and word suggestion features could enhance the quality of generated stories, while multilingual support would broaden the game's accessibility to non-English speaking users. Voice input and output capabilities could add a new dimension of interactivity, especially for users with visual impairments. Social media integration would enable users to share their stories with friends, boosting the game's visibility and fostering community engagement. Scoring systems, leaderboards, and story customization options could introduce elements of competition and personalization, enhancing user investment in the game.

APPENDICES

APPENDIX A - SOURCE CODE

```
def mad_libs():
# Prompt the user for words
adjective1 = input("Enter an adjective: ")
noun1 = input("Enter a noun: ")
verb1 = input("Enter a verb: ")
place1 = input("Enter a place: ")
plural_noun1 = input("Enter a plural noun: ")
adjective2 = input("Enter another adjective: ")
noun2 = input("Enter another noun: ")
noun3 = input("Enter one more noun: ")
adjective3 = input("Enter one more adjective: ")
verb2 = input("Enter another verb: ")
verb3 = input("Enter one last verb: ")

# Create the story with placeholders
story_template = f"""
Once upon a time, there was a {adjective1} {noun1} who loved to {verb1} all day long.
Every day, the {noun1} would {verb1} in the {place1}, hoping to find {plural_noun1}.
One day, a {adjective2} {noun2} appeared and offered the {noun1} a {noun3}.
The {noun1} was so {adjective3} that it immediately {verb2} with joy.
From that day on, the {noun1} and the {noun2} became best friends and {verb3} together every day.
"""

# Print the completed story
print("\nHere is your Mad Libs story:")
print(story_template)

# Run the Mad Libs game
if __name__ == "__main__":
mad_libs()
```

APPENDIX B -SCREEN SHOTS