LEXIS ACCTUALIZATION WORDLE

FINAL ASSIGNMENT REPORT

BASIC PROGRAMMING

By:

Muhammad Zaki (25031554091)

Muhammad Akmal Abiyu (25031554100)

Lady Puspa Khansa Yuasa (25031554173)

Lecturer :

Hasanuddin Al-Habib, S.Si.,M.Si

Siska Puspitaningsih, S.Kom.,M.Kom

STUDY PROGRAM DATA SCIENCE

FACULTY OF MATHEMATICS AND NATURAL SCIENCES

STATE UNIVERSITY OF SURABAYA

DECEMBER, 2025

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Background

In today's era of Society 5.0, a basic understanding of programming is crucial. Students are required to not only understand the theory but also be able to implement these concepts in real-world applications. We chose this simple Wordle game project because it combines fundamental programming concepts like variables, branching, looping, and functions, while also providing a fun way to train logical thinking. The title of our basic programming project is **LEXIS** which in Greece (Yunani) means word or speech, referring to nouns, verbs, adjectives, and adverbs.

1.2 Formulation of the Problem

1. How do you create a simple Wordle game using basic programming concepts?
2. How does the program handle guessing and provide feedback to the player?
3. How do you display the results and game status informatively?

1.3 Writing Purpose

1. Develop a simple Wordle game application.
2. Apply the concepts of variables, branching, looping, and functions.
3. Practice students' logic and programming skills.

CHAPTER II

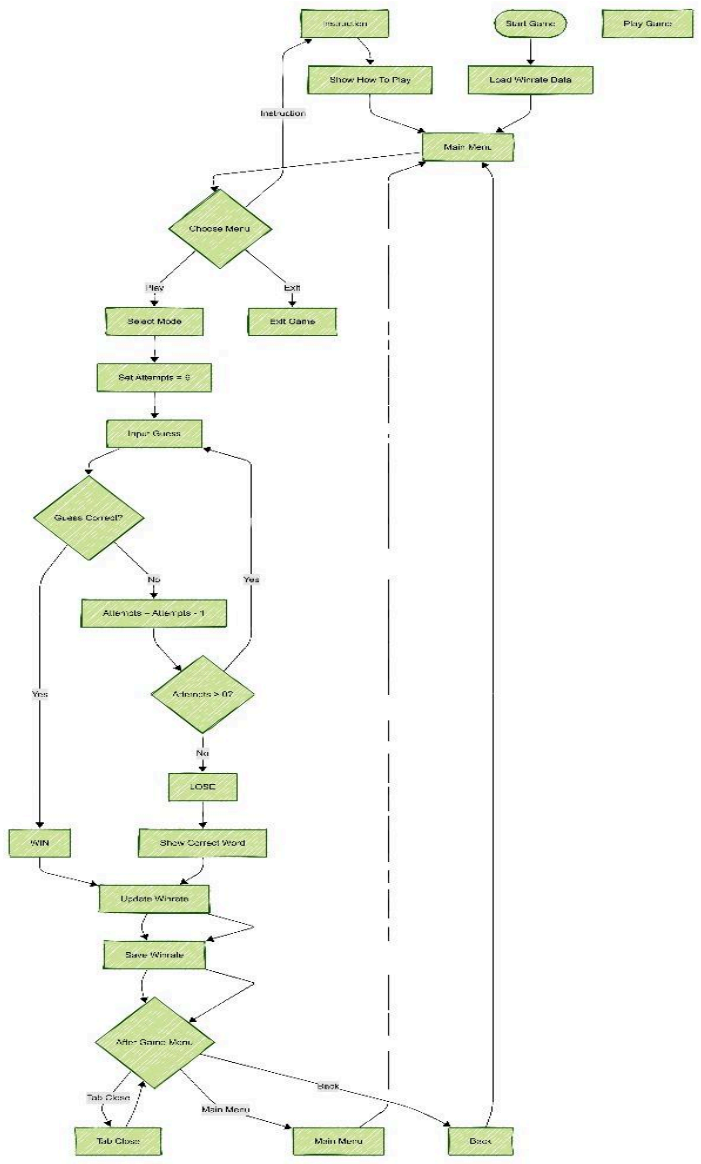
ANALYSIS AND DESIGN

2.1 Application Requirements Analysis

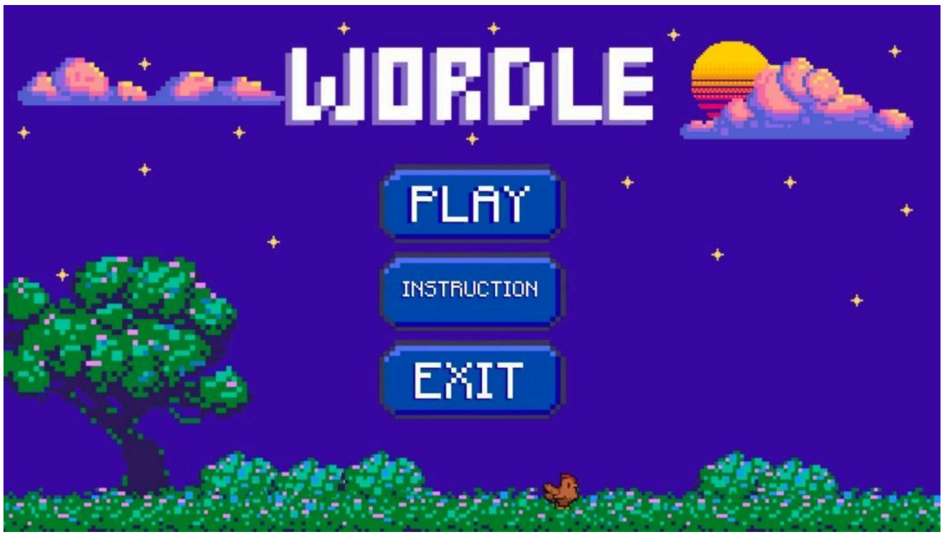
1. The program accepts input from the player to guess words.
2. It limits the number of letters guessed to a maximum of 6.
3. It provides feedback in the form of correct letters, incorrect positions, or complete errors.
4. It does not use an external database.

2.2 Flow Chart

Start -> Show Menu -> Select Play -> Input Guess -> Check Guess -> Show Results -> Win/Lose -> Finish. See the flow chart below of :



2.3 Interface Design Sketch



.

CHAPTER III

IMPLEMENTATION

3.1 Program Code Explanation

The Wordle program is built using Python and the Tkinter library for a graphical interface. The program is divided into several main modules:

1. MainMenu
   - Sets the appearance of the start menu, including the Play, Instructions, and Exit buttons.
   - Handles the menu's background music through SoundManager
   - The start_game function starts WordleApp.
   - The show_instructions function displays a pop-up containing the game rules.
   - The exit_game function closes the application.

```
class MainMenu:

    def __init__(self, root):

        self.root = root

        self.root.title("Wordle Menu")

        self.root.attributes("-fullscreen", True)

        def exit_fullscreen(event=None):

            self.root.attributes("-fullscreen", False)

        self.root.bind("<Escape>", exit_fullscreen)


        self.sound = SoundManager()

        self.sound.play_menu_music()
```

2. WordleApp
   - Handles all the logic of the Wordle game.
   - Sets up a 6x5 grid to display the guessed letters.
   - Creates a virtual keyboard and handles input from a physical keyboard.
   - Sets letter coloring based on the guess status.
   - Saves and loads win/loss statistics.

```
# load words

        try:

            with open("wordlist_upd.txt", "r") as f:
```

```
                        self.words = [w.strip().upper() for w in f if
len(w.strip()) == 5]
        except Exception:
            self.words = ["APPLE", "MANGO", "BERRY", "GRAPE", "LEMON"]


        self.secret = random.choice(self.words)
        self.wordle = Wordle(self.secret)
        self.current_guess = ""
        self.revealing = False


        self.tiles = []
        self.key_buttons = {}



        self.btn_back_img = tk.PhotoImage(file="images/back.png")
                 self.back_btn = self.bg_canvas.create_image(100, 50,
image=self.btn_back_img)
                  self.bg_canvas.tag_bind(self.back_btn, "<Button-1>",
self.back_menu)
        self.root.bind("<Escape>", self.back_menu)


        self.create_grid()
        self.create_keyboard()
        self.create_input_events()
        self.grid_frame.lift()
        self.keyboard_frame.lift()
```

```
    def update_stats_label(self):

        wins = self.stats.get("wins", 0)

        losses = self.stats.get("losses", 0)

        total = wins + losses

        percentage = (wins / total * 100) if total > 0 else 0


           display_text = f"Wins: {wins} | Losses: {losses}\nWin Rate:
{percentage:.1f}%"


                        self.bg_canvas.itemconfigure(self.stats_text_id,
text=display_text)
```

3. Wordle Class

   Set up the game logic, including the secret word, number of tries, and method for scoring guesses.

```
class Wordle:

    MAX_ATTEMPTS = 6

    WORD_LENGTH = 5

    VOIDED_LETTER = "*"


    def __init__(self, secret):

        self.secret = secret.upper()

        self.attempts = []


    def attempt(self, word):
        self.attempts.append(word.upper())



    def guess(self, word):

        word = word.upper()

        result = [LetterState(x) for x in word]

        remaining_secret = list(self.secret)
```

3.2 Description of Important Parts the Program Code

Here is a detailed description of the important parts of the program and their main roles:

1. MainMenu (Primary Menu Program)

- Acts as the entry point of the application.
- Provides options for Play, Instructions, and Exit.
- Handles the transition to the main game (WordleApp) and displays instructions or exits the application.
- Manages background music in the menu through SoundManager.

2. WordleApp (Main Game Program)

- Serves as the core engine of the game and GUI.
- Sets up a 6x5 grid to display the player's guesses.
- Creates a virtual keyboard and handles input from the physical keyboard.
- Determines the color coding of letters based on the guess results.
- Handles result popups (win, lose, invalid word warnings).
- Updates and saves win/loss statistics.

3. Wordle Class (Game Logic)

- Responsible for validating guesses.
- Stores the secret word (secret) and the history of attempts (attempts)
- Determines whether the player has solved the word (is_solved) and tracks remaining attempts (remaining_attempts).
- The guess method evaluates each letter in the player's guess and assigns its status.

4. LetterState Class (Letter Status)

Stores the status of each letter: whether it is in the correct position (is_in_position) or in the word but in the wrong position (is_in_word).

5. SoundManager Class (Sound Effects)

- Controls background music and sound effects (win, lose, menu)

- Enhances the player’s interactive experience.

6. Popup Handling (Results and Instructions)

- Displays information for wins, losses, or invalid input warnings.
- Uses background images and text to make the visual interaction more engaging.

7. Input Handling

- on_key: handles physical keyboard input.
- key_press, backspace, submit: manage letter input, deletion, and submission of guesses.
- Plays a key role in processing player input so the game flow can run smoothly.

8. Statistics Management

- Loads, saves, and displays win/loss statistics in the game.
- Provides feedback on the player’s progress.

9. Word List Management

- Loads words from wordlist_upd.txt or defaults to a predefined list.
- Selects a random secret word for each game.

10. Game Reset

- Clears the grid, keyboard, and game state.
- Allows the player to start a new game without closing the application.

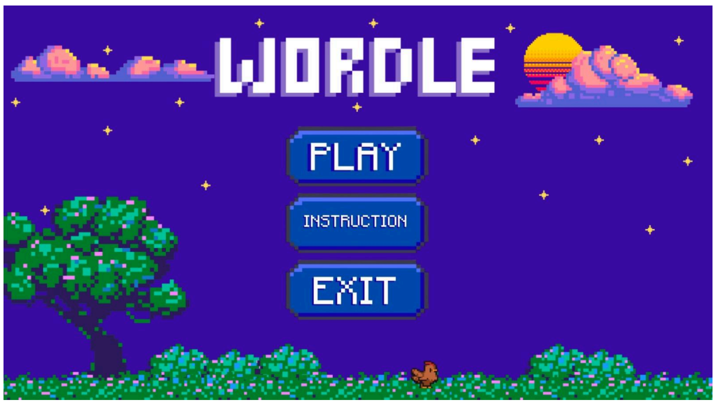
Overall, **MainMenu** serves as the entry point, while **WordleApp** and **Wordle** are the core of the game logic. These modules work together to ensure the game runs smoothly and provides an interactive experience.

3.3 Usage Manual

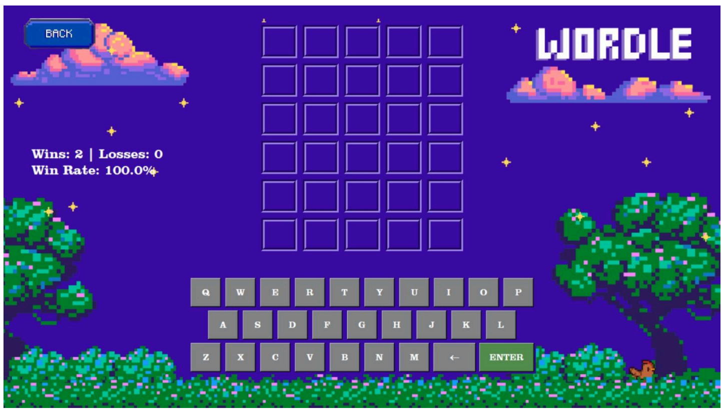
1. Run python main.py.
2. The main menu will appear; select Play to start the game.
3. Enter a 5-letter guess.
4. Note the colors that appear as clues.
5. The game ends when the word is guessed correctly or after 6 tries.
6. Win/loss statistics will be saved and displayed.
7. Use the Escape key to exit full-screen or popup mode.

3.4 Application Screenshots

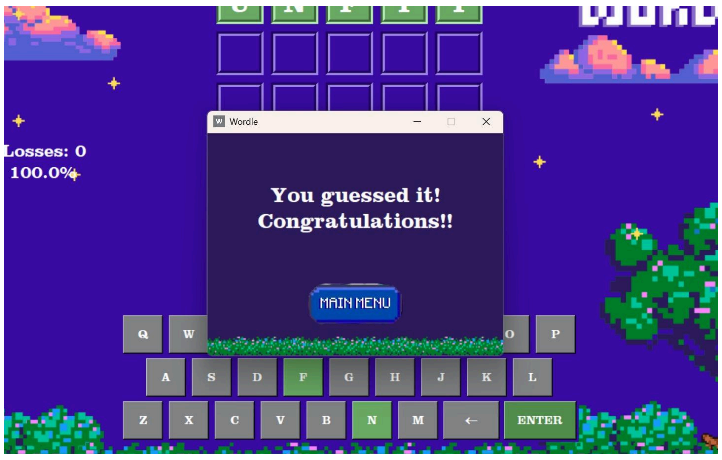
Main Menu



Keyboard and Word Grid

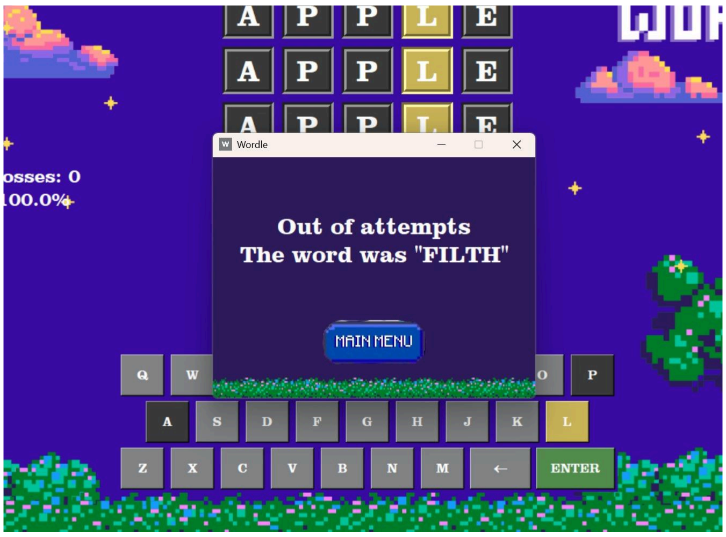


Win pop-up



You guessed it!
Congratulations!!

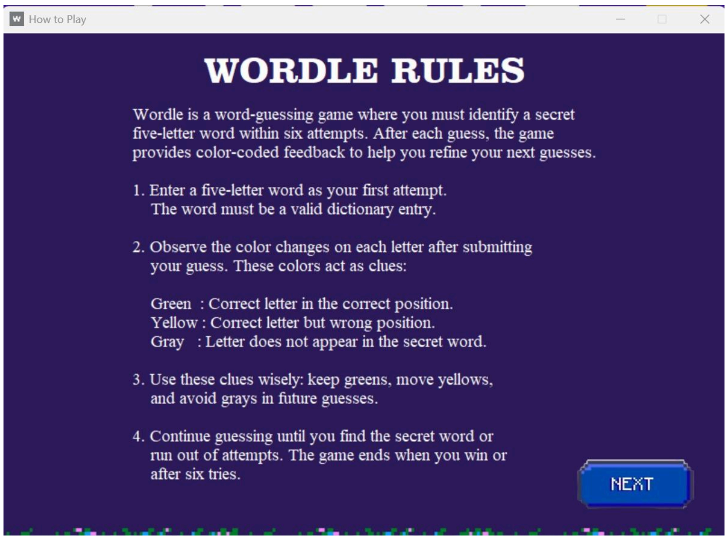
Lose Pop-Up



Wordle

Out of attempts
The word was "FILTH"

Instruction Pop-up



How to Play

WORDLE RULES

Wordle is a word-guessing game where you must identify a secret five-letter word within six attempts. After each guess, the game provides color-coded feedback to help you refine your next guesses.

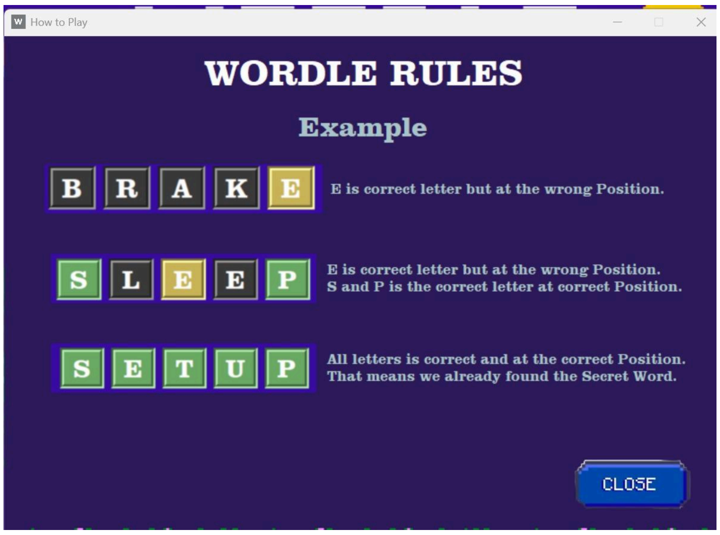
1. Enter a five-letter word as your first attempt.
   The word must be a valid dictionary entry.

2. Observe the color changes on each letter after submitting your guess. These colors act as clues:

   Green : Correct letter in the correct position.
   Yellow : Correct letter but wrong position.
   Gray   : Letter does not appear in the secret word.

3. Use these clues wisely: keep greens, move yellows, and avoid grays in future guesses.

4. Continue guessing until you find the secret word or run out of attempts. The game ends when you win or after six tries.



How to Play

WORDLE RULES

Example

B R A K E — E is correct letter but at the wrong Position.

S L E E P — E is correct letter but at the wrong Position.
S and P is the correct letter at correct Position.

S E T U P — All letters is correct and at the correct Position.
That means we already found the Secret Word.

ATTACHMENT

https://github.com/paluraheya/wordle

REFERENCE

https://youtu.be/SyWeex-S6d0?si=kDxcSph0qDZUNPOy, accessed on November 17, 2025 in YouTube