Wordle Clone

Team 10

Joe Hummer, jmhummer@ncsu.edu

Ben Morris, bcmorri4@ncsu.edu

Nick Sanford, nksanfor@ncsu.edu

Nick Schauer, njschaue@ncsu.edu

NOTE: Your final documents should **NOT** contain any << >>'s or any comments or suggestions given for each section.

Introduction: **Wordle Clone**

Wordle is a simple five-letter word guessing game that rose to popularity in 2021. During its viral rise it even caught The New York Times Company eye who acquired it to challenge its millions of subscribers, and the general public, to guess the five-letter 'Wordle' of the day. Today, it is a daily routine for some to guess the word and see how long of a streak they can hold. The goal of this comprehensive exercise is to create a similar clone to the popular word guessing game.

The program will select a random word from a list of five-letter words for the player to guess. The player will then be given six tries to guess the word. After every guess, each letter is marked by green, yellow or gray: green indicating that the letter is correct and in the correct position, yellow indicating the letter is in the word but in the wrong position, and gray indicating that the letter is not in the word at all. Each five-letter guess must be a word otherwise the program will prompt the player to try again before accepting the guess and marking any letters.

This clone will be made for two players. Each player will have 5 rounds to try and guess a random five-letter word. If a player successfully guesses the word they will be awarded points based on how many guesses it took (more points for less guesses). The player with the most points at the end of the 5 rounds wins the game.

Software Requirements

Program Start

You will use the following commands to compile and run the Wordle program:

- `$javac Wordle.java`
- `$java Wordle`

Input files (*hard coded*):

- **WordleList.txt** - The list of five-letter words to randomly select the Wordle. This list has inappropriate words removed from it.
- **GuessList.txt** - The list of five-letter words to confirm a correctly spelled English language word.

Start Menu:

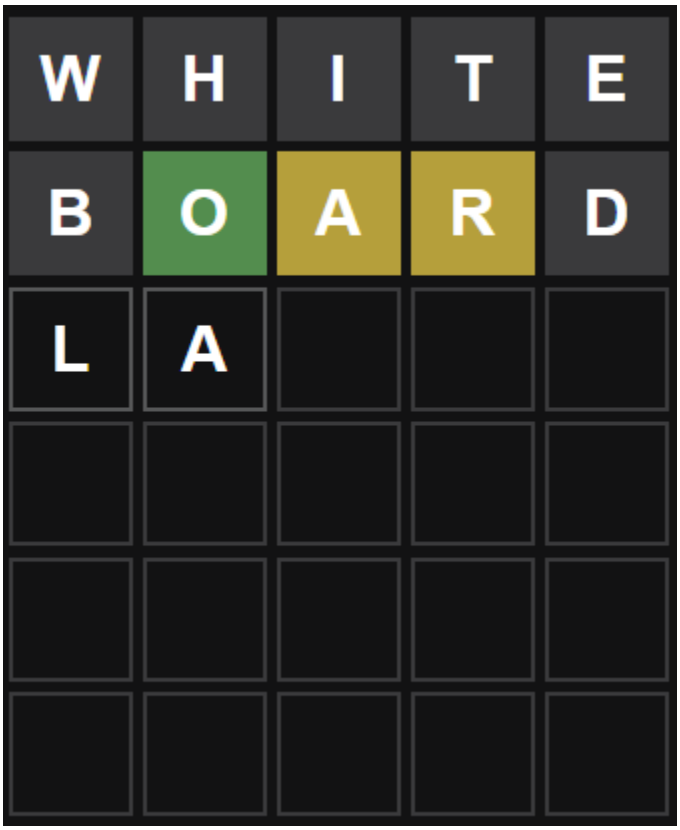
- The game will launch to the GUI below.
- Top Grid:
    - Player 1 Score: 0
    - Player 2 Score: 0
    - Round 1 of 5
    - Click Enter to Start
- The guessing grid will be ‘blank’ with all the tiles black with no text.
- The input box will be ‘blank’.

GUI / User Interface

Below is the general layout composed of 3 sections:

Player 1: ###	Player 2: ###
Round # of 5	
[Message area]	

Top Grid: Scores & Message Area



Middle Grid: Guesses

[Input area for letter]	Enter
Backspace	Quit

Bottom Grid: Inputs & Quit

Gameplay/GUI

- **Top Grid: Scores & Message Area**
  - Scores
    - Will always display player 1 and 2 scores a game goes on.
  - Rounds
    - Displays the current round you are in.
  - Message Area (and or Prompts)
    - “Click Enter to Start” - how to start the game

        - “Player #: [message]” - whos turn it is and a message if prompted
            - “Guess the wordle” - Start of turn
            - “Not a word, try again.” - word check
            - “Correct! Click Enter to Continue.” - correct guess
            - “Out of guesses. Click Enter to Continue”
        - “Game over. Player # Wins!” OR “Game over. It is a TIE!”
- **Middle Grid: Guesses**
    - Shows letters guessed and changes colors based on the letter indicator.
- **Bottom Grid: Inputs & Quit**
    - Letter input area (for each letter).
    - Enter button - move forward based on message prompt or enter letter for guess
    - Backspace button - delete guessed letters (that are not locked in)
    - Quit button - quit game

Rules

- This is a two player game.
- The game has five rounds.
- Each round, each player will guess at a Wordle.
- Each round, each player will score points based on the scoring system below.
- Each Wordle will be unique from the WordleList (no Wordle can be randomly selected more than once).
- Each player gets six guesses at their Wordle.
- Each guess must be a valid five-letter word (check against the GuessList).
- After each guess, if it is not the Wordle, the color of the tiles/letters will change to show how close your guess was to the Wordle.
    - **Green** indicates the letter is in the word and in the correct spot.
    - **Yellow** indicates the letter is in the word but in the wrong spot.
    - **Gray** indicates the letter is not in the word in any spot.
    - Letters guessed will be indicated by **dark gray** letter keys.
    - Letters not yet guessed will be indicated by light gray letter keys.

Score

If the player correctly guesses the Wordle, they get the associated points below based on the number of guesses needed. Zero points are awarded if the player is unable to guess the word in six tries. The cumulative score will be computed after each round for each player. The player at the end with the most points wins.

1. Correct guess on **first** guess: 60 pts
2. Correct guess on **second** guess: 50 pts
3. Correct guess on **third** guess: 40 pts
4. Correct guess on **fourth** guess: 30 pts
5. Correct guess on **fifth** guess: 20 pts
6. Correct guess on **sixth** guess: 10 pts

Software Design

Remember: You still will not be writing code at this point in the process.

<<Complete this section of the formal documentation by planning the classes, methods, and fields that will used in the software. Put the details of the userInterface design choice in this section.

Provide information about each class used in your project and how the classes are related.

- For each class that is **NOT** used to create objects (SEE Project 4 for an example), list:
    - STATIC METHOD HEADERS with javadoc to describe function, inputs, outputs
- For each class that **IS** used to create objects (SEE Project 5 Implementation Section for an example), list:
    - INSTANCE VARIABLES
    - CONSTRUCTOR()
    - INSTANCE METHOD HEADERS with javadoc to describe function, inputs, outputs

>>

Implementation

1. What programming concepts from the course will you need to implement your design that may require explanation?
    a. Briefly explain how each will be used during implementation.
2. Make sure to explain any java class methods you may have used that we did not cover in class for example:
    a. the Scanner class useDelimiter() method)
    b. specific classes from the javax.swing package used in a GUI implementation
    c. any exceptions thrown
    d. specific output formats, etc.

*See the implementation section in our projects 1-4 to get ideas on what to add to this section.

>>

Testing

System Testing

1. The System testing of the Wordle Clone program should be performed according to the SystemTestPlan_CE.pdf document.
2. List any test files required to run these tests.
3. Discuss any special testing scenario requirements (for example, test arrays with prespecified values to be used instead of random numbers while in test mode, command line arguments to run in test mode, etc).

Unit Testing

- No unit test files or documentation are required this semester, so this can be omitted.

Team Reflection

- Challenges faced/Lessons learned
    - What did the team learn during the project development process?
    - What problems arose and how did you solve them?

In addition to above team reflection that must be included in this document each team member must also complete an individual reflection (not in this document).

The following information should be included in each individual's peer review form of themselves.

- What did you like/dislike about the exercise?
- Any suggested changes/improvements?
- Add any comments/thoughts you care to share.