Wordle Project Analysis

a. Strategy Description

Word Selection Strategy:

In the game, the program selects a random secret word from the dictionary file (the_secret_words.txt) for each round.

For the solver, we start with no information about the word. The first guess is any word from the dictionary. After each attempt, we use the feedback to narrow down the possibilities.

How Feedback is Used:

If a letter is in the correct position → we know this letter is in that exact spot.

If a letter is in the word but in a different position → we know the letter exists but not in this position.

If a letter is not in the word → we remove all words containing this letter from the list of possibilities.

This way, after each guess, the possible words are filtered gradually until we reach the correct word.

Why This Approach is Effective:

This method quickly reduces the number of possible words after each guess.

Even with a large dictionary, the solver can usually find the word in fewer than 6 attempts.

b. Data Structure Justification

Data Structures Used:

We used a 2D array to store all words (words[MAX_WORDS][WORD_LENGTH+1]).

We used a boolean array (possible[]) to track which words are still possible after each guess.

Alternatives Considered:

A linked list could be used to store words and remove them after filtering, but arrays are simpler for a second-year project.

A hash table could speed up word lookup, but it is not necessary for a project of this size.

How Choices Support Strategy:

Arrays allow fast access to any word (O(1) by index).

The possible array makes it easy to filter words after each guess.

c. Complexity Analysis

Time Complexity:

Filtering Words: After each guess, we compare each word in the dictionary with the feedback → O(n * k), where n is the number of words and k is word length (5).

Next Guess Selection: We find the first possible word in the possible array → O(n).

Space Complexity:

Words Array: O(n * k) for storing all words.

Possible Array: O(n) to track possible words.

Graphs or Screenshots:

With a small dictionary (100 words), the solver usually succeeds in 3–5 attempts.

With a larger dictionary (1000 words), some words may take up to 6 attempts, but the solver is still efficient.