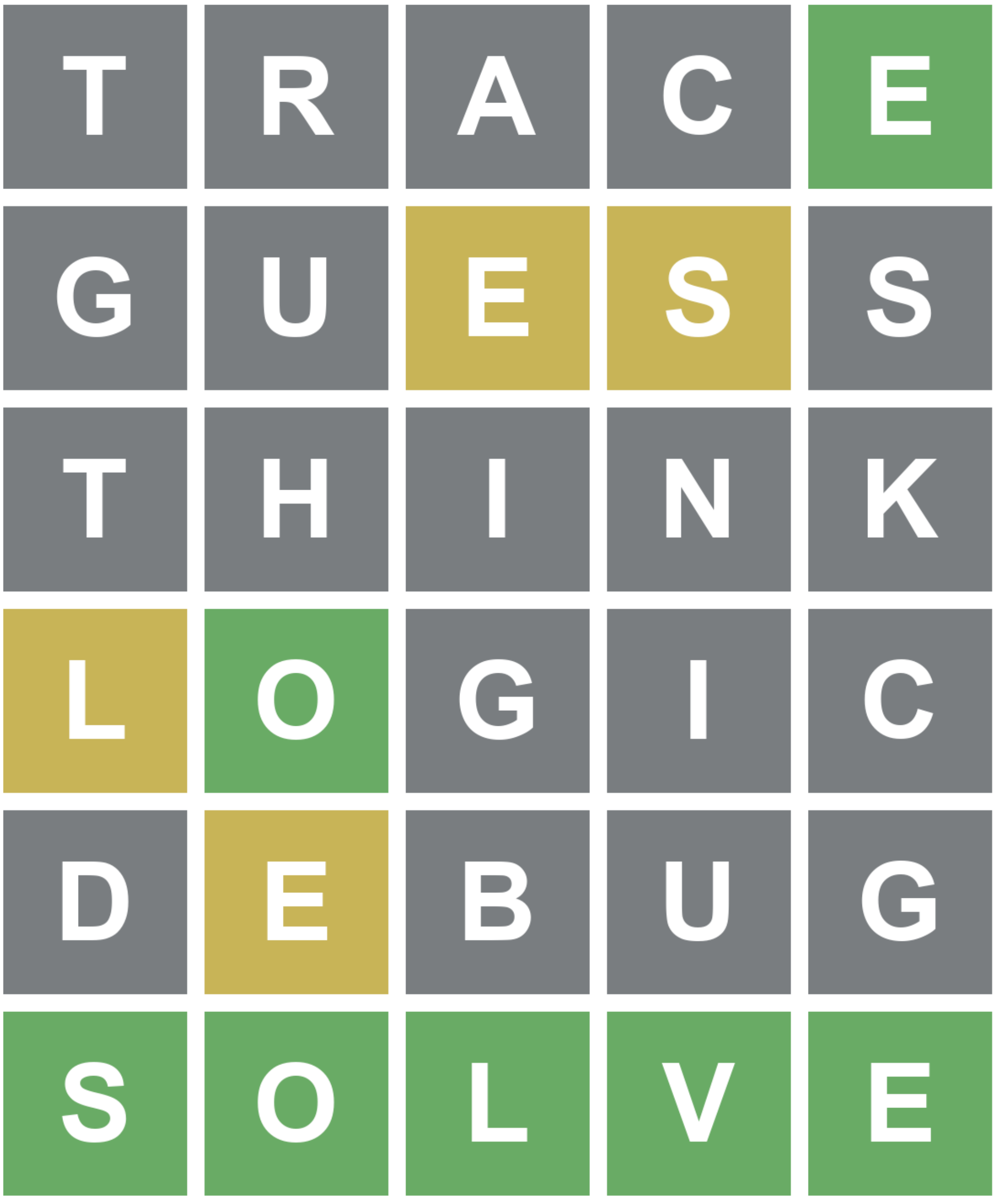
Mastering Wordle
An Algorithmic Approach

ENGR1010J 2024FA Project Manual

Teaching Team, Fall 2024



Introduction

Wordle is a popular web-based word puzzle game created by Josh Wardle. Player has at most 6 times to guess a 5-letter word. Every attempt will be marked with different colors to indicate the correctness of the guessed word. During 2023, Wordle was played 4.8 billion times.

After the success of Wordle, New York Times acquired the game for undisclosed seven-figure sum. The game has inspired many developers to create similar games and algorithms to solve them.

In this project, students will be asked to form a group to solve the Wordle game using an algorithmic approach. This project includes a programming part and a presentation part. The detailed requirements and rubrics will be provided in the following sections.

1 Programming Part

Restrictions (break any of these may lead to 0 point for all members):

1. You should only implement the project in C.
2. External open-source libraries are allowed in this project (Under Rule 1 restriction).
3. You program should be able to compile and run using gcc.
4. All the members should take part in programming. We will ask you details about your code.

Note: Start early!

1.1 Basic Wordle Game

In the provided *wordList.txt* file, there are more than 10000 5-letter words. This list consists of all the legal words for the game, i.e., if you enter a word not in this list, the program will not accept it. 2315 common words are selected from the word list to serve as the possible solutions, which will not be provided. When you enter a legal word into the program, the program will return 5 characters in the set $\{G, Y, B\}$, meaning Green, Yellow and Black. If the letter is correct, the corresponding character will be G. If the letter is correct but in the wrong position, the corresponding character will be Y. If the letter is not in the solution, the corresponding character will be B. Here is an example:

Solution	A	P	P	L	E
Guess	T	R	A	C	E
Result	B	B	Y	B	G

Some ambiguous situations may happen when there are multiple letters in the solution. If the guess is ”SPEED” and the solution is ”ABIDE”, wordle will color the first ”E” to be yellow and second ”E” to be black to indicate there is only one ”E” in the solution. To simplify the procedure, use uppercase letters all the time.

Solution	A	B	I	D	E
Guess	S	P	E	E	D
Result	B	B	Y	B	Y

More examples are listed below:

Solution	E	R	A	S	E
Guess	S	P	E	E	D
Result	Y	B	Y	Y	B

Solution	S	T	E	A	L
Guess	S	P	E	E	D
Result	G	B	G	B	B

Solution	C	R	E	P	E
Guess	S	P	E	E	D
Result	B	Y	G	Y	B

Detailed Steps

In this task, you should implement a program that can play the Wordle game. You can define the input and output format by yourself. Here is a simple example (You don't need to follow this format):

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#define MAX_WORD 15000
#define WORD_LEN 5
char wordList[MAX_WORD][WORD_LEN+1];
char solution[WORD_LEN+1];
void readWordList() {
    FILE *fp = fopen("wordList.txt", "r");
    for (int i = 0; i < MAX_WORD; i++) {
        fscanf(fp, "%s", wordList[i]);
    }
    fclose(fp);
}
void selectSolution() {
    srand(time(NULL));
    int idx = rand() % MAX_WORD;
    strcpy(solution, wordList[idx]);
}
int main() {
    readWordList();
    selectSolution();
    // Your code here
    return 0;
}
```

These are the steps you need to follow:

1. Read the word list from the file wordList.txt. The file contains more than 5-letter words.
2. Define a solution word.
3. Ask the user to guess the word.
4. Implement the game logic. (Mainly about checking the word)
5. Print the result. (5 characters in the set $\{G, Y, B\}$)
6. You may need to restrict the user to guess at most 6 times and give points (discussed in next section).

I/O Format requirements

You can design your input and output format by yourself. But in order to pass JOJ test and further algorithm design part test, you are asked to implement a library.

A library is a collection of functions that can be used in other programs. You can check lab4 or project material for help.

You will use a function pointer to pass your player function in the wordle game. (You can simply write a function randomly guess a word or ask user input). We will provide you only with the header file. And you are allowed to add more functions, but you should not change the function definition. Global values are allowed to save your life.

1.2 Algorithm Design

In this part, you should design an algorithm to solve the Wordle game. To make competition fair, we will not provide any suggestion or advice on how to solve it efficiently. You can search online for some ideas. But you should notice, our wordList is different from standard wordle game, which may make the solution different. Do not copy from internet directly.

Rubrics

We will have totally 1000 words to test your algorithm, which are selected from the standard 2315 wordle possible solution. For each game, you will receive points pf 11 - rounds. For example, if you solve the game in 3 rounds, you will get 8 points. You have totally 10 attempts to solve the problem. Under this rubric, the maximum score you can get is 10000 points (A good solution would get more than 7000, get full points is impossible unless you already know the solution directly). We will order the teams by total points and re-distribute using normal distribution. And your final score in this part will range between 40 and 100. (40 is for basic implementation)

One important thing to mention is that every group will have **at most 3 times** to submit. Program will run on TA's personal computer since this is a quite heavy task for JOJ. You should make sure your program runs perfectly. Any kind of error will be considered as 0 points. We will record the highest attempt. If all 3 attempts are less than 100, you will get 0 points in this part. Also, the score board will be available. You can check your score and ranking at any time.

The algorithm should be able to run in a reasonable time. (We will not provide the time limit, but time exceed/memory exceed/compile error and other errors will also cost one chance.) We suggest you to test your program once after the first week working to make sure you are working on this. We will give bonus points for excellent performance.

2 Presentation Part

There will be a presentation for this project. You will have 5 minutes to present your project (Mainly focus on algorithm part). You should divide your work evenly in both programming and presentation part. i.e. someone focus on programming, someone focus on presentation is not allowed.

3 Evaluation Criteria

The grading of this project will be based on the following components:

- Part 1 - Basic Wordle Game: 20% (Full score if playable, possible bonus for more features)
- Part 2 - Algorithm Design: 60% (Possible bonus for excellent performance)
- Part 3 - Project Presentation: 20% (Full score if clear and concise, possible bonus for excellent presentation)

Note: The project counts 10% in the final grade. All members in a group will receive the same score.

4 AI usage

In this project, you need to contact with your TA very closely. But our job this time is to make it clear what you need to do. We will not help you to debug or give you any hint. You are highly recommended to use AI to work with this project since it is quite complex. You can use AI to search information, come up with ideas, and then you could talk with your group members to form a solution logic.

Then you can ask AI to generate different part of the code. AI is also good at writing comments and tests. Although AI is powerful, it is not perfect. You should be responsible for the final result, and be able to explain every part of your code and algorithm.

Here is an example of how to use code to build a Wordle game (not complete, and in Python):

```python
import random
wordList = []

# Ask AI: Read wordList.txt and store it in wordList
with open("wordList.txt", "r") as f:
    wordList = f.readlines()

solution = "APPLE" # User defined solution
# Ask AI: Use a loop to play the game, set the maximum guess to 6
for i in range(6):
    guess = input("Enter␣your␣guess:␣")
    result = ""
    # Ask AI to further modify the code (current code is not correct)
    for j in range(5):
        if guess[j] == solution[j]:
            result += "G"
        elif guess[j] in solution:
            result += "Y"
        else:
            result += "B"
    print(result)
    if result == "GGGGG":
        print("You␣win!")
        break
```

There are many good AI tools. If you are using VSCode, you can try to get an Education License to use GitHub Copilot. It is a good tool to help you write code. This extension allows you to modify the code directly in VSCode, without switching to the browser or APPs. You can try to find some videos about young kids write code using Copilot. This tool can efficiently convert your natural language into code.

5 Submission

There are many timestamp for this project, we divide the work in many parts to avoid you start at last minutes.

- Part 1 - Basic Wordle Game: 2024-11-22 23:59pm
- Part 2 - Algorithm Design: 2024-11-26 23:59pm
- Part 3 - Project Presentation: 2024-11-26 23:59pm
- Final Submission: 2024-12-02 23:59pm (All test should be done)

All submission should be done on Canvas. You should submit a zip file contains everything about the project.

6 Reference Material

You may find the following resources useful for you to perform well:

1. Standard wordle game website (New York Times): https://www.nytimes.com/games/wordle/index.html
2. Unlimited wordle game: https://wordly.org
3. Possible Optimal solution: https://jonathanolson.net/experiments/optimal-wordle-solutions
4. 3B1B solution: https://www.youtube.com/watch?v=v68zYyaEmEA
5. 3B1B mistake clarification: https://www.youtube.com/watch?v=fRed0Xmc2Wg
6. Andrew Steele solution: https://www.youtube.com/watch?v=YEoCBnQwdzM
7. Other general wordle solvers: https://www.thewordfinder.com/wordle-solver/
8. Other general wordle solvers: https://word.tips/wordle/