**C++ PROJECT**

**WORDLE**

**JENNIFER FONTI**

**N1025571**

**Immagine che contiene tavolo

Descrizione generata automaticamente**

Sommario

[Analysis: 3](#_Toc101689768)

[Problem Identification: 3](#_Toc101689769)

[Research of existing problem solutions: 4](#_Toc101689770)

[Program requirements: 4](#_Toc101689771)

[Design and Implementation: 4](#_Toc101689772)

[Testing: 6](#_Toc101689773)

[Future improvements: 7](#_Toc101689774)

# Analysis:

I wanted to create a program in C++ that recreates the famous New York Times’ game *Wordle*. Wordle is a word puzzle game where you must guess a different 5 letter word each time you play. Every time that a word is inserted by the user the program checks if any of the letters is present in the guess, if it is and it’s in the right position the letter is going to become green, if it’s present but not in the right position it is going to become yellow and lastly, it it’s not present it is going to turn grey.

Immagine che contiene testo

Descrizione generata automaticamente

# Problem Identification:

The original Wordle is a web application, therefore is coded mainly in JavaScript (back end) and HTML/CSS (front end). While the JavaScript back end could be easily rethought in C++, the front-end part comes more difficult to “parse” using any C++ GUI library.

At the very beginning, while I was analysing the project, I opted to use Qt library to develop the GUI, but I found many problems between the open source version of Qt and my application.

After that, I tried numerous graphic libraries without much success, some for incompatibility with my computer some because they required purchases to use basic components.

In the end, I decided to give a try to Visual Studio 2019, using a .net framework for the front end and even if with multiple difficulties it ended up being the right choice for my application.

# Research of existing problem solutions:

As I said before, the most problematic part was the graphic. In fact, it required a lot of research to find the correct GUI library.

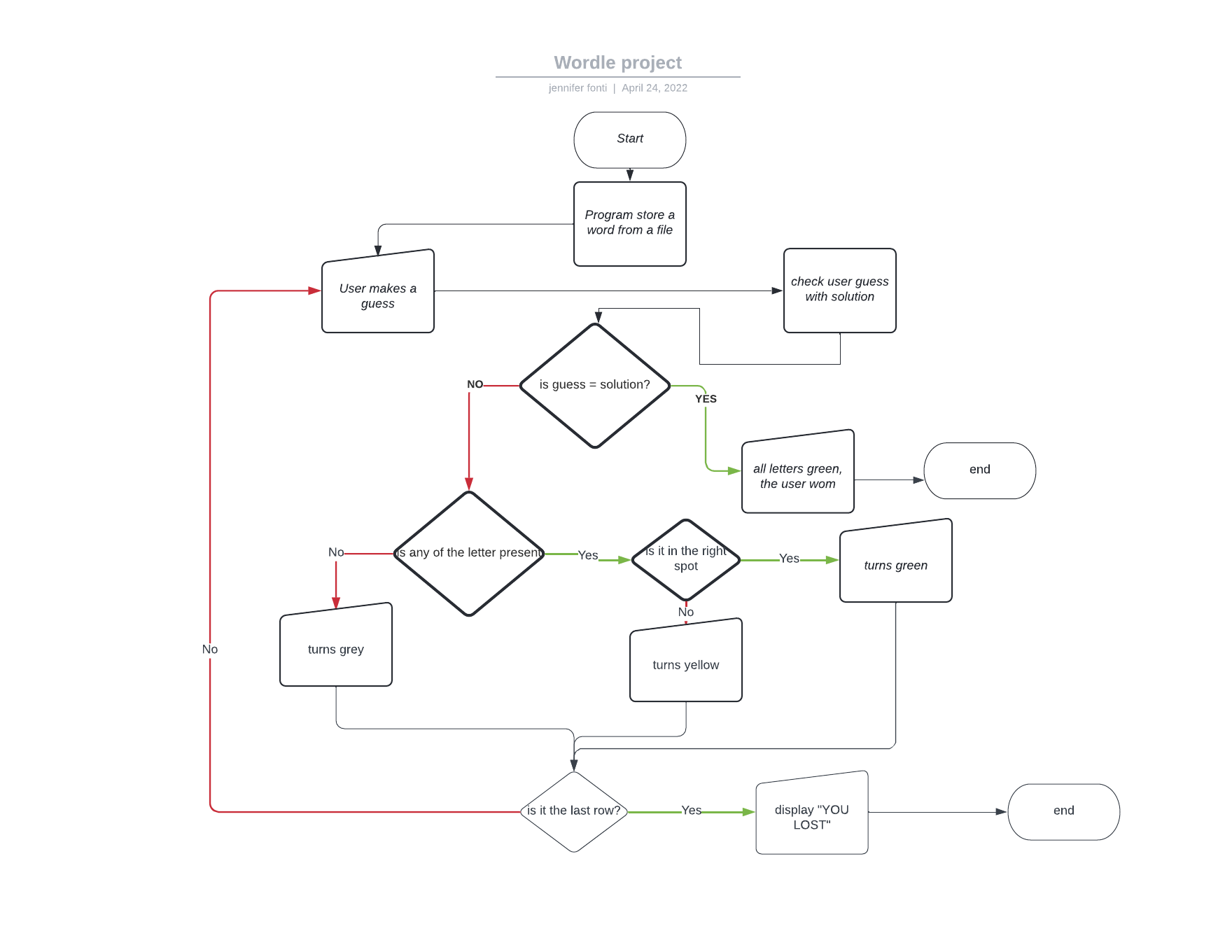
When I chose Visual Studio, I wasn’t strongly convinced it would have worked so I started following numerous You Tube tutorials on how to use a .net framework. Once I had the knowledge, I needed I started developing the application but being my first time coding something so complex in C++ I came along some errors. I found the solutions to most of the problems on the Microsoft Visual Studio help web page and a few other on Stack Overflow.

# Program requirements:

The requirements for this program should include a valid Graphic User Interface, a file handling part to change the solution word every time that the game is restarted and lastly, being object oriented.

After much research I manged to meet all the requirements stated above.

# Design and Implementation:



This is how I implemented the main part of my program:

1. The program reads and save in an array a file that contains some of the most 5 letter English words present in the dictionary.
2. Then it choses randomly one word out of the whole array
3. The user inserts a word, the program checks if it is a 5 letter one, if not doesn’t allow the user to go to the second row
4. If a 5 letters word is detected after the user pressed enter the program checks if the letters are in the right position and assign every letter a colour.
5. If the word is corrected it stops the game, and displays the text: “you won”
6. If not, it keeps repeating the loop (point 3 to 5) till it gets to the last row
7. If the user doesn’t guess the word by the end of the last row it displays the word, and the user lost the game.

The main problem that I came across during this process was the impossibility to make a ref class (.net managed class) and a normal C++ class communicate properly.

To solve this, I created a class “Wrapper” that parsed my normal class code to a manged class, so I was able to call its methods from the main window.

I found some other less significant problems while developing the project such as the impossibility if using some of the native c++ types in the .net framework, but I managed to overcome these problems using different solutions.

# Testing:

1. User insert a word:

Immagine che contiene tavolo

Descrizione generata automaticamente

1. Winning case:

Immagine che contiene tavolo

Descrizione generata automaticamente

1. Losing case:

Immagine che contiene tavolo

Descrizione generata automaticamente

# Future improvements:

I am planning on improve this project by creating a diagram showing the statistics of the user, like percentage of winning and losing games and how many times the user got the solution for each row

I am also thinking about change the background-color of each key in keyboard when used in the grid, using the same logic as the text Boxes.