```
pub fn submit_guess(&mut self, guess: String) -> bool {
    if guess.len() != 5 {
        println!("{}", "your guess must be 5 letters long".red().italic())
    } else if self
        .possible_words
        .iter()
        .find(|v| **v == guess.to_lowercase())
        .is_none()
    {
        println!("{}", "invalid word, try again".red().italic())
    } else {
        self.guesses.push(guess.to_lowercase());
        if guess.to_lowercase() == self.correct_word.to_lowercase() {
            println!("{}", guess.to_lowercase().green().bold());
            self.has_won = true;
            return true;
        }
        let guess_chars = guess
            .to_lowercase()
            .chars()
            .collect::<Vec<char>>();
        let correct_chars = self.correct_word
            .chars()
            .collect::<Vec<char>>();

        let mut final_str: String = String::new();

        let mut chars_found: Vec<char> = vec![];

        for i in 0..=4 {
            let guess_char = guess_chars[i];
            let correct_char = correct_chars[i];

            if guess_char == correct_char {
                chars_found.push(guess_char.clone());
                final_str = format!(
                    "{}{}",
                    final_str,
                    guess_char.to_string().green().bold()
                );
            } else if correct_chars.iter().find(|v| {
                    v == &&guess_char
                }).is_some() && correct_chars
                    .iter()
                    .fold(0, |acc, v|
                        if v == &guess_char {
                            acc + 1
                        } else {
                        acc
                        }
                    ) > chars_found
                        .iter()
                        .fold(0, |acc, v|
                            if v == &guess_char {
                                acc + 1
                            } else {
                                acc
                            })
            {
                chars_found.push(guess_char.clone());
                final_str = format!(
                    "{}{}",
                    final_str,
                    guess_char.to_string().yellow().bold()
                );
            } else {
                final_str = format!(
                    "{}{}",
                    final_str,
                    guess_char.to_string().black()
                );
            }
        }

        println!("{}", final_str);
    }
    false
}
```