Wordle Cracker: Unlocking Possible Winning Words with String Matching Algorithm and Regular Expression

Dhanika Novlisariyanti - 13521132
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13521132@std.stei.itb.ac.id

***Abstract*—This document is created to suggest a few winning words regarding the game wordle. Wordle is a web-based guessing word game in where user can try up to six times to guess the given word that day. Wordle can only be played once a day, that is what makes this game popular as you must try it the next day to win this game. This document will attempt to give all possible winning words through regular expression and Knuth-Morris-Pratt Algorithm.**

Keywords—wordle, regular expression, levenshtein-distnce, algorithm, matching letter, function

I. INTRODUCTION

Wordle is a worldwide web-based game created by Welsh software engineer Josh Wardle. Later then, it was bought by the New York Times Company in 2022, and ever since then the game became popular. A lot of people played it in their spare time. The catch is you can only play this game once a day, that's what makes it fun. Players will attempt to guess a five-letter word within six attempts. The game started out blank without any given hint or information. People will usually make the first guess a word which contains all vowels as many words contain a lot of vowels. When sending the answer, Wordle will give hint from the first guess. If the word of the day contains that letter but is not in the correct position, it will give a yellow color. As the player guesses the correct letter and position, it will give a green color. Wordle can be very tricky, as words in English have the same suffix but same prefix such as catch, batch, match, patch. The author picks this topic seeing that the author loves to play this game but not proficient enough in English words. Sometimes the author cannot recognize the given English words due to familiarity and not really used in daily life. That is why this is the author's attempt to make a list of possible winning words from Wordle with regular expression and Knuth-Morris-Pratt Algorithm.

Source: https://www.nytimes.com/games/wordle/index.html

II. THEORITICAL FOUNDATION

A. *Regular Expresion*

Regular Expression is a pattern that can use to match a distinct and often specific combination of characters [2]. Regular Expressions are helpful to inspect and process strings input. Regular Expression is an object that can be created using regular expression literals (//) or invoking the regular expression constructor function using new keyword [2]. Example:

Figure 2.A.1 Creating New Regular Expression

The most basic thing in regular expression is finding literal string. Example when finding "rat" in a word it will match with a paragraph that contains the word "rat". If there are more than one match, it will adjust to how many "rat" words there are. Another thing you can do in regular expressions are metacharacter. Metacharacter are specialized characters that affect the process of finding the pattern in a text [4]. You can use character class to only specify certain letter to match in a text such as

Construct	Description
[abc]	a, b, or c
[^abc]	All characters except a, b c
[a-zA-Z]	a until z or A until Z, inclusive (range)

[a-d[m-p]]	a until d or m until p
[a-z&&[def]]	d,e, or f
[a-z&&[^bc]]	a until z, except b and c
[a-z&&[^m-p]]	a until z, and not m until p

Table II.A.1. Regular Expression

Regular Expression also provides a quantifier to define the number of pattern repetition. If the pattern wants to match only on certain position, regular expression provides boundary matchers.

Construct	Description
X?	X appears one time or not at all
X*	X appears zero or more
X+	X appears one or more
X{n}	X appears exactly n times
X{n, }	X appears at least n times
X{n,m}	X appears n to m times

Table II.A.2. Regular Expression

Regular Expressions can make pattern easier to read and reduce mistakes by using predefined character class.

Construct	Description
.	All characters
\d	Digit[0-9]
\D	Non digit [^0-9]
\s	Whitespace character
\S	Non whitespace character
\w	Word character[a-zA-X_0-9]
\W	Non word character

Table II.A.3. Regular Expression

These are some examples of usage in regular expressions. By using these, pattern matching will be much easier. All these usages can be accessed through the internet.

B. Levenshtein Distance Algorithm

Levenshtein Distance Algorithm is very impactful because it does not require text to be in the same length. It is invented in 1965 by Vladimir Levenshtein, a soviet Mathematician [5]. Informally, the Levenshtein distance between two words is the minimum number of single-character edits (insertions, deletions or substitutions) required to change one word into the other [5]. The larger the output distance implies that more changes were executed so that the two words are equal.

$$\operatorname{lev}(a,b) = \begin{cases} |a| & \text{if } |b| = 0, \\ |b| & \text{if } |a| = 0, \\ \operatorname{lev}\left(\operatorname{tail}(a), \operatorname{tail}(b)\right) & \text{if } a[0] = b[0] \\ 1 + \min \begin{cases} \operatorname{lev}\left(\operatorname{tail}(a), b\right) \\ \operatorname{lev}\left(a, \operatorname{tail}(b)\right) \\ \operatorname{lev}\left(\operatorname{tail}(a), \operatorname{tail}(b)\right) \end{cases} & \text{otherwise,} \end{cases}$$

Figure 2.B.1 Levenshtein Distance in Mathematical Form

C. Wordle

Wordle is a web-based online puzzle word game in which players will guess the word of the day. Players will be able to guess the five-letter word within six attempts. Players will be given hint after each guess by the colors indicated in each square. Gray squares means that there is no letter in the word. Yellow squares means that there is that letter in the word but unfortunately the position of the letter is wrong. As for green squares, it means you have guessed the correct position and letter.

Figure 2.C. 1 Wordle

The web itself have a simple interface as to why it is so popular. Another factor that adds to how popular Wordle is players can only play it only once a day. Despite how game is intended to be addicting and meant to be played again and again, but wordle can only be played once a day. That is why a lot of people keep updated with Wordle as each day a new word is different and guessing it will be a challenge.

III. PROBLEM DECOMPOSITION

In this chapter, the author will explain how this program works. This program is constructed in python using regular expression library as well as English dictionary library. Since the Wordle that we will find all winning words is in English. The program will be filtering through the English dictionary by using regex by finding all the included words and the ones that are not included. After getting all possibles words from the dictionary in an array, if the player guesses something that the word is in order or not, the author provided a Levenshtein-Distance Algorithm distance algorithm to use to match the word that is sequentially has the same pattern.

A. *Regular Expression Function*

Based on the user input, the letter that is included in the words that the user wants to search it will be put in to "included" variable as to the not included will be put into the "not included" variable. Then the letter will be processed by the regular expression defined below

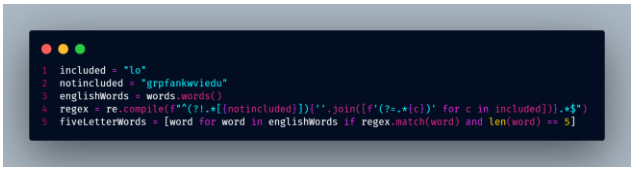

Figure 3.A.1 Regular Expression Function

Construct	Description
^	This will match the start of the text
(?!.*[{notincluded}])	This is to ensure that the string will not contain any letter that the user input. It uses {notinclude} because it waits for the user input.
''.join([f'(?=.*{c})'for c in included])	This is to ensure that the string will contain all the letter that the user input. The opposite of the notincluded function
.*	This will matches any characters except new line
$	This will match the end of the string

Table III.A.1. Regular Expression Function

B. *Dictionary Matching*

Once it passes through the regular expression function, all possible patterns that contain all the included words and not the included words will be match through dictionary. This is to ensure that the words that will be outputted are real words that exist in the dictionary.

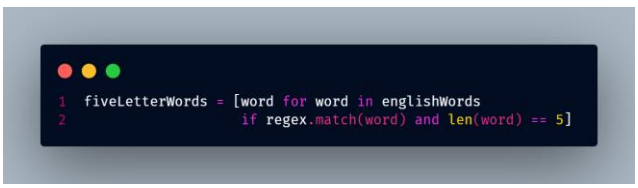

Figure 3.B.1 Wordle

Not only with the matching word, but it also ensures that the word length is five words. This makes it easier since wordle is a five-letter word puzzle. This will be such a hassle and need a lot more constraint if the words are not a fixed length.

C. *Levenshtein Distance Match*

When the user is given the array output of all the possible words from the included and not included words, it still has a lot of words. Another approach is to use brute force or greedy algorithm because right there it all depends on luck whether that word is useful for the next clue or not at all. To reduce the amount to brute force all the possible words considering players are only given six attempts of try, the author gives another approach by using levenshtein distance.

Typically, unless the players have immense good luck, the first attempt is never correct. Player can guess a good word by getting yellow squares, or all grey squares. If a player gets all grey squares in the first attempt, it is guaranteed that the letter will not appear. So, the player will not guess a word with those letters again.

In the second guess, if a player has a lot of yellow squares. It is better for the player to guess the correct position from all the possible letters and check it. Another option is by having another filler word to guess if another letter is present. By using a filler word, players chance is either get a lot of grey boxes, or another yellow boxes or possibly a green box. If it is a good hint, and the player is not dumb enough to use a word that is already checked then the chances of guessing the word are higher.

By the third guess, the player can now start using regular expression and levenshtein distance. As by the third guess, enough information should be gathered to list all the possible words. But if by third guess, the player still does not get a yellow or green box, the author suggests using another filler word to try narrow the list of all possible words. A good filler word would be a word that contains vowels, and by the second and third guess it does not contain the same letter. The author personally suggests using "GROUP", "FLANK", "WIVED". Those three words contain all vowel A, I, U, E, O, and there are no repetitive letters. There are other set of filler words, it is best to decide when guessing the word.

This approach hopefully will make the player guess it in the third or fourth try. By listing all the possible words from the dictionary, the player will try to find which word is suitable for the next guess. Reducing the stress in thinking of all the possible words, this program will do all the thinking and hopefully with this, the player will get words that are not even thinkable in the moment. Wordle answers are sometimes English words that are rarely used or usually involved a repetition of a letter. For the latter, for non-native English speaker who enjoys playing Wordle sometimes cannot think of a possible word even if all the clues are all laid out. This program since it will be matched by the English dictionary, will list out all the possible words. For the repetitive letter, wordle clues usually are different. Example, if the answer is ENTER, and the player guessed the word EVERY, it will give the green box for the first letter of E and a yellow box of E. But what happens if a player guessed the word STARE, it would give a yellow box for T and E, but it does not specify which E. This is why sometimes it is frustrating for the player to guess because there are too many possibilities.

```python
def levenshtein_distance(pattern: str, data: List[str]) -> List[str]:
    """Measures the minimum number of single-character edits
    (insertion, deletions, or substitutions)"""
    match = []
    pattern_length = len(pattern)
    pattern_lower = pattern.lower()
    best_match = []

    for item in data:
        question_length = len(item)
        distance = [[0] * (question_length + 1) for _ in range(pattern_length + 1)]

        for i in range(pattern_length + 1):
            distance[i][0] = i

        for j in range(question_length + 1):
            distance[0][j] = j

        for i in range(1, pattern_length + 1):
            for j in range(1, question_length + 1):
                if pattern_lower[i - 1] == item[j - 1]:
                    distance[i][j] = distance[i - 1][j - 1]
                else:
                    distance[i][j] = min(
                        distance[i - 1][j],
                        distance[i][j - 1],
                        distance[i - 1][j - 1]
                    ) + 1

        percentage = (1 - distance[pattern_length][question_length] /
                      max(pattern_length, question_length)) * 100
        if percentage >= 30:
            best_match.append({"text": item, "percentage": percentage})

    best_match.sort(key=lambda x: x["percentage"], reverse=False)

    for item in best_match:
        match.append(item["text"])

    return match
```

Figure 3.C.1 Levenshtein Distance Function

This levenshtein function will return an array of all the words from highest to lowest changes from the given list of fiveLetterWords.

To test this, the author will use wordle unlimited since the wordle real game can only be used once a day.

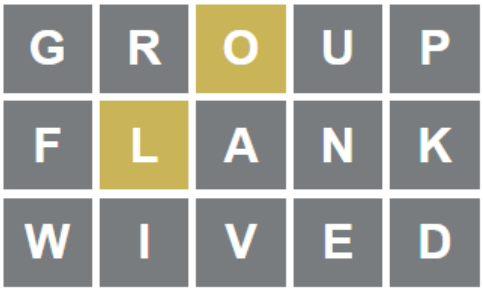

Figure 3.C.2 Wordle

As the picture displayed, the player has attempted to guess three words, but the only clue is there is an “L” and “O” somewhere in the words. And from the clue given, it looks like that “LO” will not be in the second and fourth place. From the regular expressions function, we get these words.

```
[Running] python -u "d:\00_STEI ITB\04_SMT4\STRATEGI ALGORITMA\MAKALAH\wordleCracker.
py"
['bloom', 'bolly', 'booly', 'Cholo', 'clomb', 'cloot', 'closh', 'cloth', 'cohol',
'colly', 'cooly', 'coyly', 'coyol', 'hollo', 'Holly', 'holly', 'hooly', 'hotly',
'jhool', 'jolly', 'jolty', 'lobby', 'lochy', 'lolly', 'looby', 'lotto', 'Molly',
'molly', 'mools', 'octyl', 'oolly', 'Polly', 'Rollo', 'scolb', 'shool', 'sloom',
'slosh', 'sloth', 'smolt', 'sotol', 'stool', 'stylo', 'tolly', 'tolyl', 'xylol',
'zloty', 'cloth']
```

Figure 3.C.3 Result from Regular Expression Function

As you can see, there are a lot of options from the regular and we only have three attempts left. This is going to be a problem. This is where the levenshtein distance function came in handy. As we gather clues, it is stated that “LO” is not in the second and third position. So, we insert “LO” to the levenshtein distance as it will give us a list of words that contains lo and sort in the way how many time that the word changes.

```
['bloom', 'Cholo', 'clomb', 'cloot', 'closh', 'cloth', 'hollo', 'lobby', 'lochy',
'lolly', 'looby', 'lotto', 'Rollo', 'sloom', 'slosh', 'sloth', 'stylo', 'xylol',
'zloty', 'cloth']
```

Figure 3.C.4 Wordle

From the given words, the list is already reduced from 43 elements to 19 words. All the words are now coming down to a narrow size possible and thinkable words since it is displayed, and the player can think what to choose to guess the next word. Now the player can try to analyze all these words. Since “L” and “O” are not in the second and third position, “BLOOM”, “CLOMB”, “CLOOT”, “CLOTH”, “SLOOM”, “SLOSH”, “SLOTH”, “ZLOTY” is off limits.

Figure 3.C.5 Wordle

For some reason, the player tried to guess “LOBBY” and unexpectedly it is the correct words. This is the attempt where the player guessed it correctly. But sometimes, it does not go the way it should be. Since this program does not receive any information from the wordle website, it will not be a hundred percent accurate and relies on all the clues that are given. The best outcome would be in the third guess or fourth guess. The worst outcome would be not guessing the word at all. This is possible if the word contains a lot of repetitive words, but a long-time player would usually be able to guess when the word is containing a repetitive letter, such as when guessing words, all possible letter keeps getting grey box. That’s when it comes to guessing a word that contains a repetitive word. This program is created to help the player list all the possible words and not give the exact winning word as the program is not connected to the web and cannot hack it to get all the information.

IV. CASE STUDY

Based on the explanation from the previous chapters, here we will test out some of the cases and analyze at most how many attempts it takes to guess the word by the help of regular expressions and levenshtein distance.

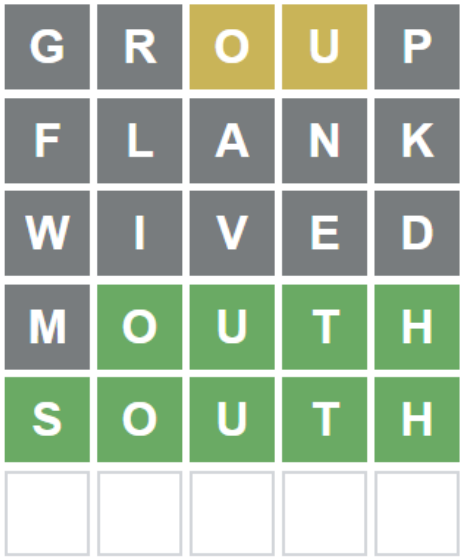

Figure 4.1 Test Case 1

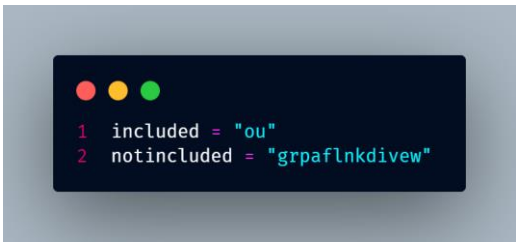

Figure 4.2 Test Case 1

['Aotus', 'bousy', 'bouto', 'Bucco', 'bumbo', 'buxom', 'byous', 'chous', 'Cobus', 'Comus', 'couch', 'couth', 'cusso', 'hocus', 'housy', 'hucho', 'humbo', 'husho', 'jocum', 'joust', 'jumbo', 'justo', 'Kobus', 'Momus', 'mousy', 'mouth', 'Notus', 'ocuby', 'outby', 'quoth', 'scout', 'shout', 'smous', 'smout', 'South', 'south', 'stout', 'thuoc', 'totum', 'touch', 'tousy', 'tsubo', 'youth', 'mouth', 'south', 'touch']
['Aotus', 'bousy', 'bouto', 'byous', 'chous', 'Cobus', 'Comus', 'couch', 'couth', 'hocus', 'housy', 'jocum', 'joust', 'Kobus', 'Momus', 'mousy', 'mouth', 'Notus', 'ocuby', 'outby', 'scout', 'shout', 'smous', 'smout', 'South', 'south', 'stout', 'totum', 'touch', 'tousy', 'youth', 'mouth', 'south', 'touch']

Table 4.1 Test Case 1

For this attempt, it takes five guesses. From the first guess, the given clue is that there is “O” and “U” somewhere in the word, but it is not in the third and fourth position. As the player keeps guessing, the player already gathered enough clues to start guessing the correct word. From the given list, we need to find the word without OU, and it is filtered by the second array. The player tried to guess with “MOUTH” but alas it is not correct. It narrows down to “YOUTH, “SOUTH”, and “TOUCH” and there are only two attempts.

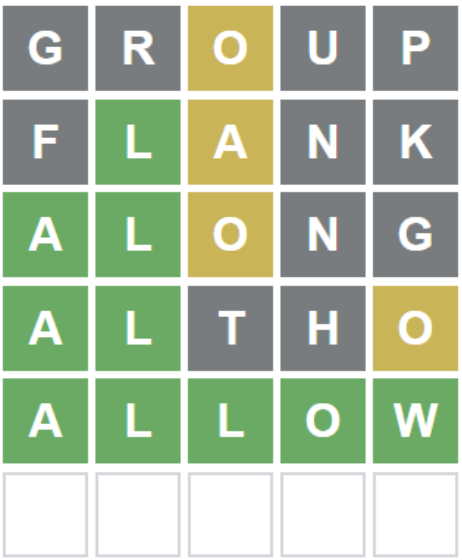

Figure 4.3 Test Case 2

Figure 4.4 Test Case 2

['abilo', 'ablow', 'aboil', 'acold', 'alamo', 'aldol', 'aliso', 'allot', 'allow', 'alloy', 'alody', 'aloed', 'aloid', 'aloma', 'Alosa', 'alose', 'alowe', 'altho', 'amole', 'amylo', 'atelo', 'atoll', 'azole', 'Balao', 'balao', 'baloo', 'balow', 'bloat', 'bocal', 'bowla', 'boyla', 'bozal', 'callo', 'Chola', 'chola', 'cloam', 'coaly', 'Colla', 'colza', 'comal', 'cowal', 'coxal', 'dobla', 'dolia', 'domal', 'dotal', 'Eloah', 'Falco', 'Goala', 'Haloa', 'haole', 'holia', 'holla', 'idola', 'joola', 'lacto', 'lasso', 'loach', 'loamy', 'loath', 'loave', 'lobal', 'local', 'loxia', 'loyal', 'maleo', 'Malto', 'modal', 'molal', 'oadal', 'oasal', 'oliva', 'omlah', 'osela', 'ossal', 'Pablo', 'Paola', 'Polab', 'Rotal',

'Salmo', 'Salol', 'salol', 'salvo', 'shoal', 'shola', 'solay', 'Solea', 'solea', 'Somal', 'somal', 'stola', 'talao', 'tlaco', 'total', 'viola', 'vocal', 'Volta', 'volva', 'votal', 'woald', 'zoeal']
['abilo', 'ablow', 'aboil', 'acold', 'alamo', 'aldol', 'aliso', 'allot', 'allow', 'alloy', 'alody', 'aloed', 'aloid', 'aloma', 'alose', 'alowe', 'altho', 'amole', 'amylo', 'atelo', 'atoll', 'azole', 'Balao', 'balao', 'baloo', 'balow', 'bocal', 'bozal', 'callo', 'coaly', 'comal', 'cowal', 'coxal', 'domal', 'dotal', 'Falco', 'Goala', 'Haloa', 'haole', 'lobal', 'local', 'loyal', 'maleo', 'Malto', 'modal', 'molal', 'oadal', 'oasal', 'ossal', 'Pablo', 'Paola', 'Rotal', 'Salmo', 'Salol', 'salol', 'salvo', 'shoal', 'Somal', 'somal', 'talao', 'total', 'vocal', 'votal', 'woald', 'zoeal']

Table 4.2 Test Case 2

This is the type of word that contains repetitive letters. From the clue given, “AL” is in the correct position, but “O” is not. From the fourth and fifth guess, it gives clue that “O” is in the fourth position. From the filtered list, the player can check which word starts with “AL” and “O” in the fourth position.

Figure 4.5 Test Case 3

['Andre', 'ender', 'trend']
['Andre', 'ender', 'trend']

Figure 4.6 Test Case 3

Table 4.3 Test Case 3

This testcase generates a short array of possible words. From the given clue, “ND” seems to be the ending letter and “R” is in the second position. From the given array, ”TREND” is the only possible answer.

Figure 4.7 Test Case 4

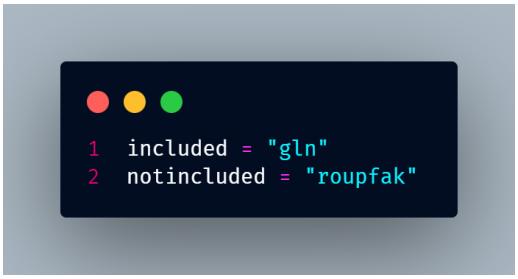

Figure 4.8 Test Case 4

['cling', 'glent', 'glint', 'ingle', 'Kling', 'ligne', 'linge', 'lingy', 'lying', 'sling']
['glent', 'glint', 'ingle']

Table 4.4 Test Case 4

This is one of the attempts where the player gets to guess it in the third attempt. From the first word, “G” is somewhere in the letter. From the second word, “L” is somewhere in the word and “N” is in the correct position. With the added leveinshtein distance, we can reduce the possible words and remove the word with “GL” in the front.

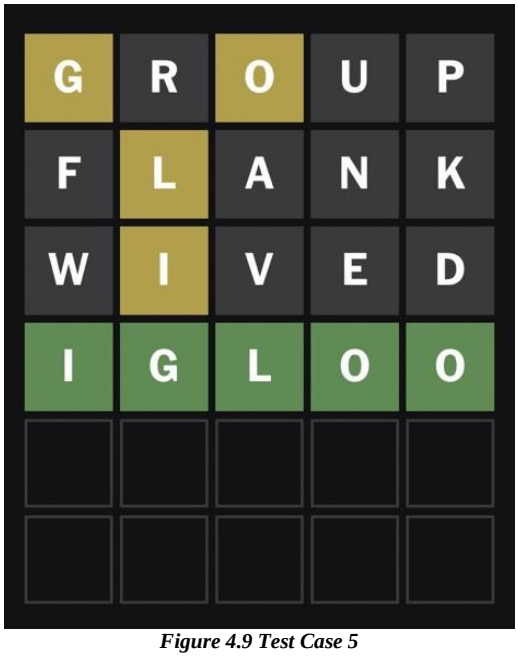

Figure 4.9 Test Case 5

Figure 4.9 Test Case 5

['Algol', 'bogle', 'dogly', 'eloge', 'globe', 'globy', 'glome', 'gloom', 'gloss', 'glost', 'glove', 'gloze', 'godly', 'goldy', 'golee', 'golem', 'Golgi', 'golly', 'goloe', 'gools', 'goyle', 'igloo', 'lodge', 'logic', 'logie', 'logoi', 'logos', 'Molge', 'ology', 'segol', 'glove']
['Algol', 'bogle', 'dogly', 'eloge', 'godly', 'goldy', 'golee', 'golem', 'golly', 'gools', 'goyle', 'ology', 'globe', 'globy', 'glome', 'gloom', 'gloss', 'glost', 'glove', 'gloze', 'goloe', 'igloo', 'glove']

Table 4.5 Test Case 5

V. CONCLUSION

Based on the test case provided in chapter IV, it takes most of the guesses to be correct in the fourth try. From statistics, the chance to guess Wordle on the first try is 0.043% considering only possible answers and 0.008% considering allowed guesses [6]. According to data collected from Wordle players, the most common number of attempts required to guess the correct word is 4 (33.10% of games), followed by 5 attempts (23.91% of games) and 3 attempts (22.66% of games). Players cannot guess the correct word in 2.92% of games [6].

Usually, the first and second guess is to provide clues whether what kind of word further guesses need to be guessed. From the study case, most of it guesses on the third try or fourth try. This is rising the attempt to guess the word and the chance of winning are higher.

ACKNOWLEDGMENT

This task was created to fulfill IF2211 Strategy Algorithm. Thanks to Bu Ulfa for teaching this whole semester in my class. Thanks to Pa Rinaldi for creating all the presentations material needed to study this subject. Thanks to my friend Nasywa for helping me in doing the study case for Wordle.

REFERENCES

[1] T.Bennet,"Wordle", https://www.nytimes.com/games/wordle/index.html , May, 2023.

[2] M. Conner, "Regular Expression", https://dev.to/mconner89/regular-expressions-13jn , August 24, 2020.

[3] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[4] Wibisono Yudi, Masayu Leyli Khodra, "Modul Praktikum Kuliah: Pengantar Regular Expression", April 11, 2002

[5] Vatsal, "Text Similiraty with Levenshtein Distance in Python", https://towardsdatascience.com/text-similarity-w-levenshtein-distance-in-python-2f7478986e75 , March 14, 2022

[6] Matic Broz, "Wordle Statistics, Facts, & Strategies", https://photutorial.com/wordle-statistics/#:~:text=On%20average%2C%20Wordle%20players%20can,the%20theoretical%20best%20(0.043%25) , March 10, 2023

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 22 Mei 2023

Dhanika Novlisariyanti 13521132