CECS 174 - Project 3
“Project CECS-174 Style Wordle”
Due date: 04/14/22

Team: team_xyz

Student Name: Alfredo Regla
Student ID: 027962816

I certify that this submission is my original work

Alfredo Regla

Student Name: Joshua Correa
Student ID: 029196984

I certify that this submission is my original work

Joshua Correa

Project Report: Programming Project 3 - “Project CECS-174 Style Wordle”

1. **Goal:**
   Our goal was to create a simple version of the popular game Wordle. In Wordle it is your job to guess the secret 5 letter word after some many attempts. You can only enter REAL 5 letter word guesses. Throughout your guesses the program gives you hints about what the secret word can be. By the end you either run out of attempts and fail, or you guess the word and win!

2. **Problem Description:**
   In this program we are required to implement a system that takes a secret word, then checks if it is valid in respect to being a real word and a 5-letter word. After that we take a player word that is then evaluated in a similar way. After that we compare the secret word and player word to see if they are the same. If they have similar elements but are not the same we display that to the player. After we compare we repeat the process of comparing new player words and the secret word however many attempts the user imputed. After the player runs out of guesses or guesses the word the program ends.

3. **Program Description:**
   a.
      For our problem we needed to first ask the user to input a secret 5 letter word. If the secret word is longer than 5 letters or isn't a real word the player is asked to

input the secret word again; if the word is true we store it in the variable is_word. After the secret word is imputed and tested we then ask the user to input the number of attempts and store it in the variable N. Once we establish an amount of attempts, the next player (the one guessing the secret word) inputs their guess as to what the secret word is; this variable is saved as player_word. Once their guess is filtered through a similar word checker the secret word was placed through, the player word and secret word is compared. If the secret word and player word have the same character in the same position the program relays that information to the user, and the variable letters_in_the_right_position is increased by one. If the secret word and player word have the same character in a different position the program relays that information to the user. If the secret word and player word have different characters nothing happens. Once the player word and secret word are compared you go to your next attempt and the process repeats by the player entering a new player word every attempt. If during those attempts the player gets the right word the program congratulates the player and ends. If after those attempts the player still hasn't gotten the word the player loses and the program ends.

b.

Welcome to Wordle - CECS 174 edition!
Enter the secret 5-letter word: tough
Input allowed number of attempts: 1
Enter your attempt #1
round
You entered a 5-letter word
o is in the secret_word and in the correct spot #2
Correct letters in the correct spot: 1
u is in the secret_word and in the correct spot #3
Correct letters in the correct spot: 2
You already used #1 attempts. Better luck tomorrow!

We chose this particular test case in order to test if our program could produce the correct number of attempts used at the end of the game.

Welcome to Wordle - CECS 174 edition!
Enter the secret 5-letter word: asdfghjk
Not a valid word, try again!
Enter the secret 5-letter word: banana
Not a valid word, try again!
Enter the secret 5-letter word: scooby doo

Not a valid word, try again!
Enter the secret 5-letter word: skill
Input allowed number of attempts: 1
Enter your attempt #1
qwertyu
Not a valid word, try again
Enter your attempt #1
lumbago
You entered a 7-letter word, but a 5-letter word is needed. Try Again.
Enter your attempt #1
ice cream
Not a valid word, try again
Enter your attempt #1
skill
You entered a 5-letter word
s is in the secret_word and in the correct spot #1
Correct letters in the correct spot: 1
k is in the secret_word and in the correct spot #2
Correct letters in the correct spot: 2
i is in the secret_word and in the correct spot #3
Correct letters in the correct spot: 3
l is in the secret_word and in the correct spot #4
Correct letters in the correct spot: 4
l is in the secret_word but not in the correct spot
l is in the secret_word but not in the correct spot
l is in the secret_word and in the correct spot #5
Correct letters in the correct spot: 5
Congrats you won using 1 attempt(s)

We also have this test case where we tested the word checkers capability for both the secret word and the player word.

Welcome to Wordle - CECS 174 edition!
Enter the secret 5-letter word: ocean
Input allowed number of attempts: 10
Enter your attempt #1
plane
You entered a 5-letter word
a is in the secret_word but not in the correct spot
n is in the secret_word but not in the correct spot
e is in the secret_word but not in the correct spot

Enter your attempt #2
skill
You entered a 5-letter word
Enter your attempt #3
skull
You entered a 5-letter word
Enter your attempt #4
comma
You entered a 5-letter word
c is in the secret_word but not in the correct spot
o is in the secret_word but not in the correct spot
a is in the secret_word but not in the correct spot
Enter your attempt #5
cough
You entered a 5-letter word
c is in the secret_word but not in the correct spot
o is in the secret_word but not in the correct spot
Enter your attempt #6
lumbago
You entered a 7-letter word, but a 5-letter word is needed. Try Again.
Enter your attempt #6
asdsfdgf
Not a valid word, try again
Enter your attempt #6
banana
You entered a 6-letter word, but a 5-letter word is needed. Try Again.
Enter your attempt #6
rough
You entered a 5-letter word
o is in the secret_word but not in the correct spot
Enter your attempt #7
though
You entered a 6-letter word, but a 5-letter word is needed. Try Again.
Enter your attempt #7
double
You entered a 6-letter word, but a 5-letter word is needed. Try Again.
Enter your attempt #7
ocean
You entered a 5-letter word
o is in the secret_word and in the correct spot #1
Correct letters in the correct spot: 1

c is in the secret_word and in the correct spot #2
Correct letters in the correct spot: 2
e is in the secret_word and in the correct spot #3
Correct letters in the correct spot: 3
a is in the secret_word and in the correct spot #4
Correct letters in the correct spot: 4
n is in the secret_word and in the correct spot #5
Correct letters in the correct spot: 5
Congrats you won using 7 attempt(s)

Finally we chose this test case in order to put our program through the test of longevity and ensure that it properly loops and detects different entries.

Welcome to Wordle - CECS 174 edition!
Enter the secret 5-letter word: tough
Input allowed number of attempts: 0

We chose this test case in order to test our program's capability for detecting a number of attempts input that was not a positive number.

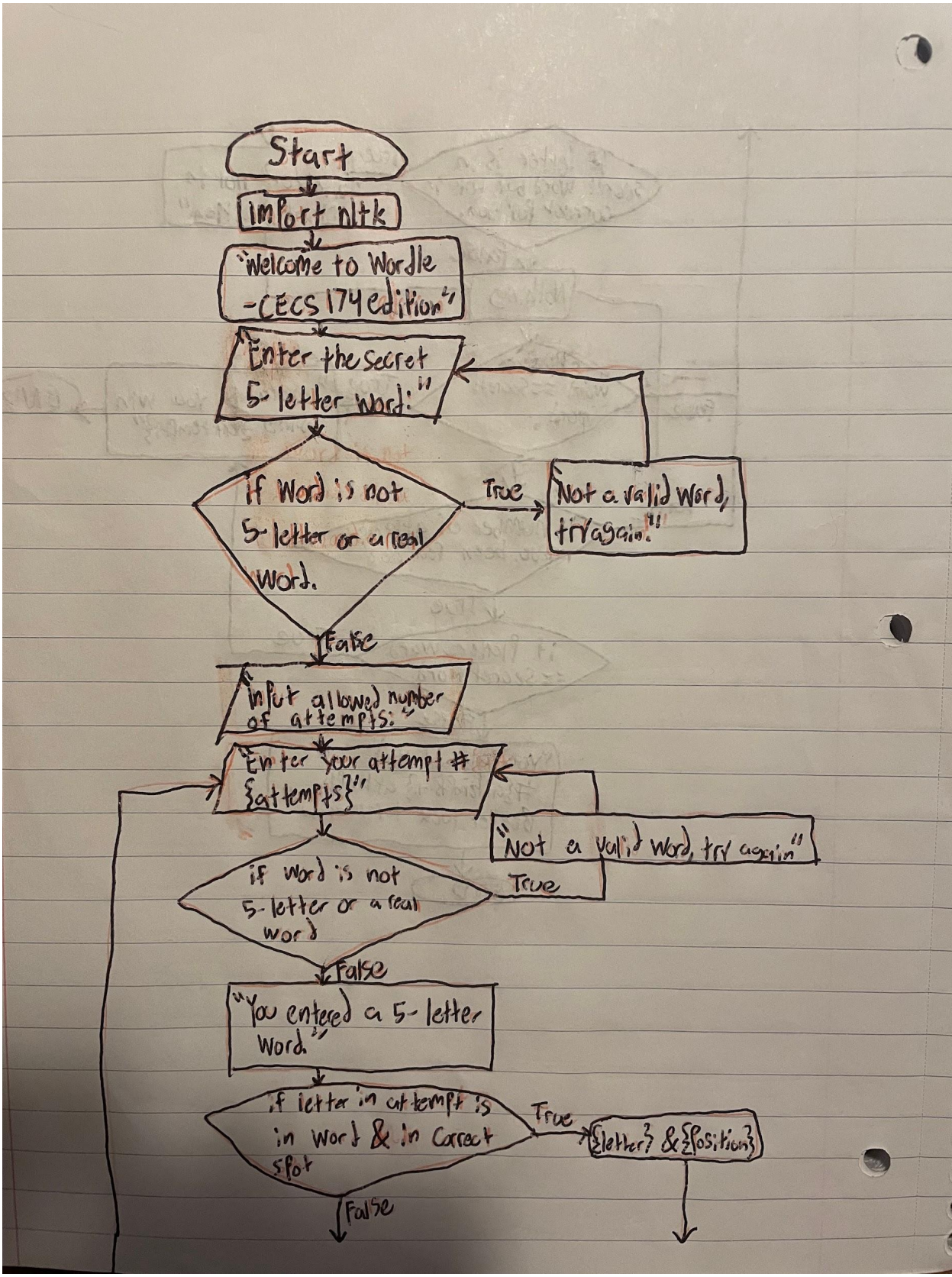
Welcome to Wordle - CECS 174 edition!
Enter the secret 5-letter word: 78
Not a valid word, try again!
Enter the secret 5-letter word: tough
Input allowed number of attempts: 2
Enter your attempt #1
345
Not a valid word, try again
Enter your attempt #1
tough
You entered a 5-letter word
t is in the secret_word and in the correct spot #1
Correct letters in the correct spot: 1
o is in the secret_word and in the correct spot #2
Correct letters in the correct spot: 2
u is in the secret_word and in the correct spot #3
Correct letters in the correct spot: 3
g is in the secret_word and in the correct spot #4
Correct letters in the correct spot: 4
h is in the secret_word and in the correct spot #5
Correct letters in the correct spot: 5

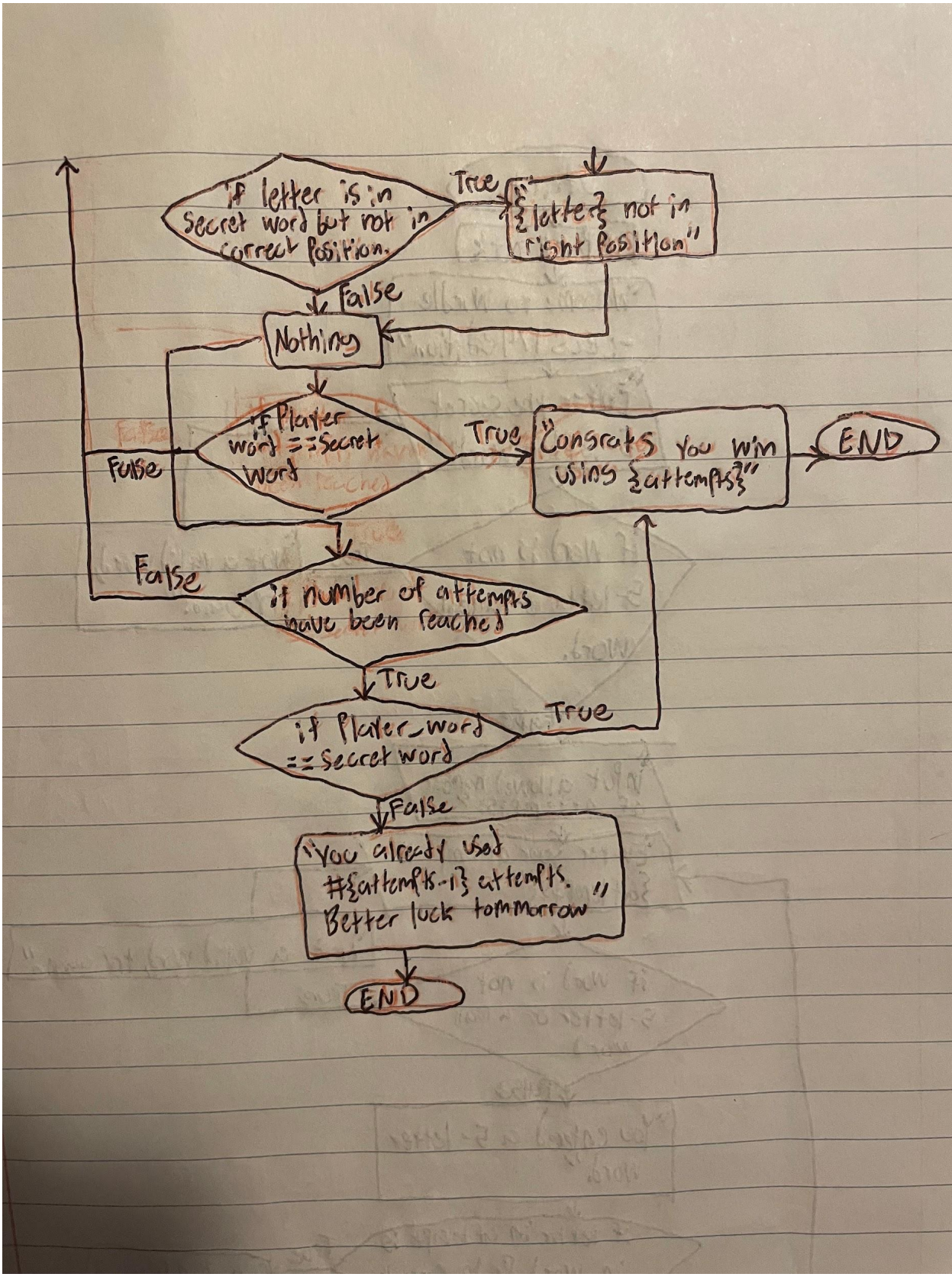
Congrats you won using 1 attempt(s)

We chose this particular test case in order to see if our program could tell whether an input could detect numbers as an invalid input for a secret word and player word.

c. In simple steps (pseudo code) explain the algorithm you want to use to solve the problem for any input
When player number 1 enters a word we want to first check if the word is real and if the word is 5 characters long; if the word is not real and/or is not 5 characters long we want to ask the user to input the word again. After player 1 enters a valid secret word we then ask them how many attempts they want to give player number 2. Once a number of attempts has been established we then want to ask player 2 for their guess as to what the secret word is. We then want to check the player word to see if it is real and 5 characters long, if it doesn't meet the aforementioned criteria then we ask player 2 to input another guess. When the guess is valid we compare the secret word to the player word and find similarities (same characters in same position or same characters in different positions). Once the player either runs out of guesses or the player correctly guesses the secret word, the program ends.

d. Flow chart on page 7 and 8.


Start
import nltk
"Welcome to Wordle -CECS 174 edition"
"Enter the secret 5-letter word:"
if Word is not 5-letter or a real word.
True
"Not a valid Word, try again!"
False
"Input allowed number of attempts:"
"Enter your attempt # {attempts}"
"Not a valid word, try again"
True
if word is not 5-letter or a real word
False
"You entered a 5-letter word."
if letter in attempt is in word & in correct spot
True
{letter} & {position}
False


if letter is in secret word but not in correct position.
True
"{letter} not in right position"
False
Nothing
if Player word == Secret word
True
"Congrats you win using {attempts}"
END
False
if number of attempts have been reached
False
True
if Player_word == Secret word
True
False
"You already used #{attempts-1} attempts. Better luck tommorrow"
END

4. **Program Implementation:** Describe the implementation
    a. We only used the str and int data types in this code. We used str for the initial secret word and the guesses that the user inputs, we used int for the number of attempts that the user wants. We used, !=, ==, and <. We used not equal for our while loop and used == for simple things like True and False. The last operation we used was < and that was for our main while loop which was While (attempts < N), which handled the main part of the code.
    b. What was challenging while you were implementing your program?
    The most challenging thing when implementing the program was creating the for loop to begin our code, and we figured out we had to use a nested loop to create the game and that's when we started to code everything else.
    c. The most straightforward thing to implement was creating the inputs, asking the user for a word and attempts, and creating the initial while loop. This was all in the beginning of the code.
    d. After we finished the code, we tested for words that didn't exist, we tested for words that have less than 5 letters. We tested these scenarios in each part of the code, so when you put the initial word and also for your guesses. We did this to make sure there were no bad inputs.
    e. Our program handles bad input really well , when asking for any word, if the input is anything other than a valid 5 letter word, the program tells you, 'Not a valid word, try again!'. The only time our program breaks is in the beginning
    f. Bugs and/or Errors:
        i. The only bug is that when you ask the user for the number of attempts and the input is not a number the whole code breaks.
        ii. Another bug we had then fixed was that the number of attempts was always 1 more than expected, to fix this instead of having (attempts <= N) we changed it to (attempts < N), doing this allowed the attempts to never equal N.

5. **Conclusion**
    a. What went well is that we troubleshooted the beginning of the code, when asking the initial questions and processed what was going on in each line of code. This helped us as we went through each while statements and decided what to do.
    b. I think we could have managed our time differently, as we were having trouble in the beginning when deciding how to loop the guesses, so deciding when to input the loops before could have helped.
    c. The project could have been improved by having the TA do another flowchart before we started to code as that helped a lot in project 2.

d. Some people had no idea what wordle was, so maybe play a game of wordle before coding to maybe get people interested and more intrigued rather than going in without knowing what the game is.

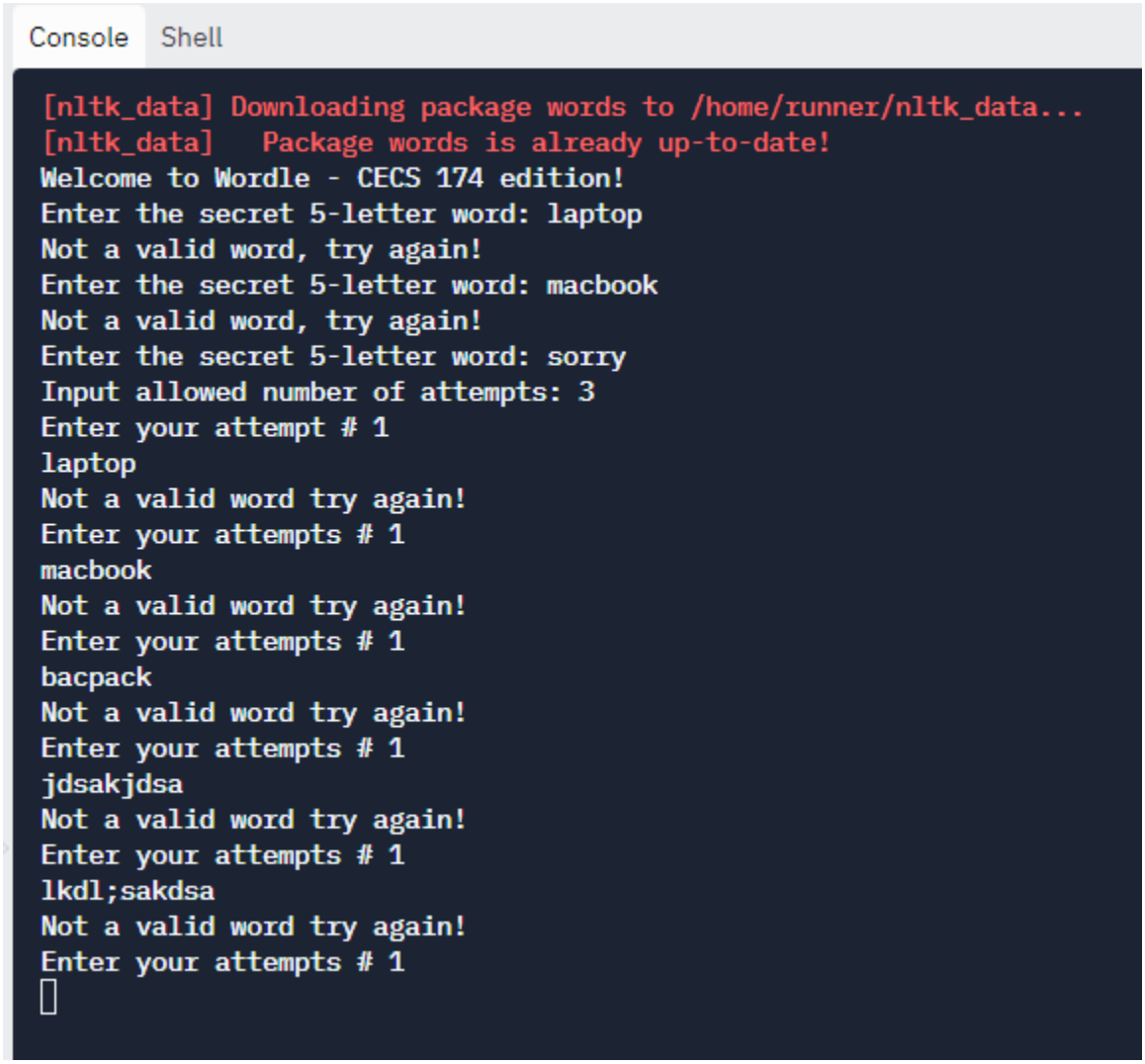
Appendix:

Project Wordle Source code:

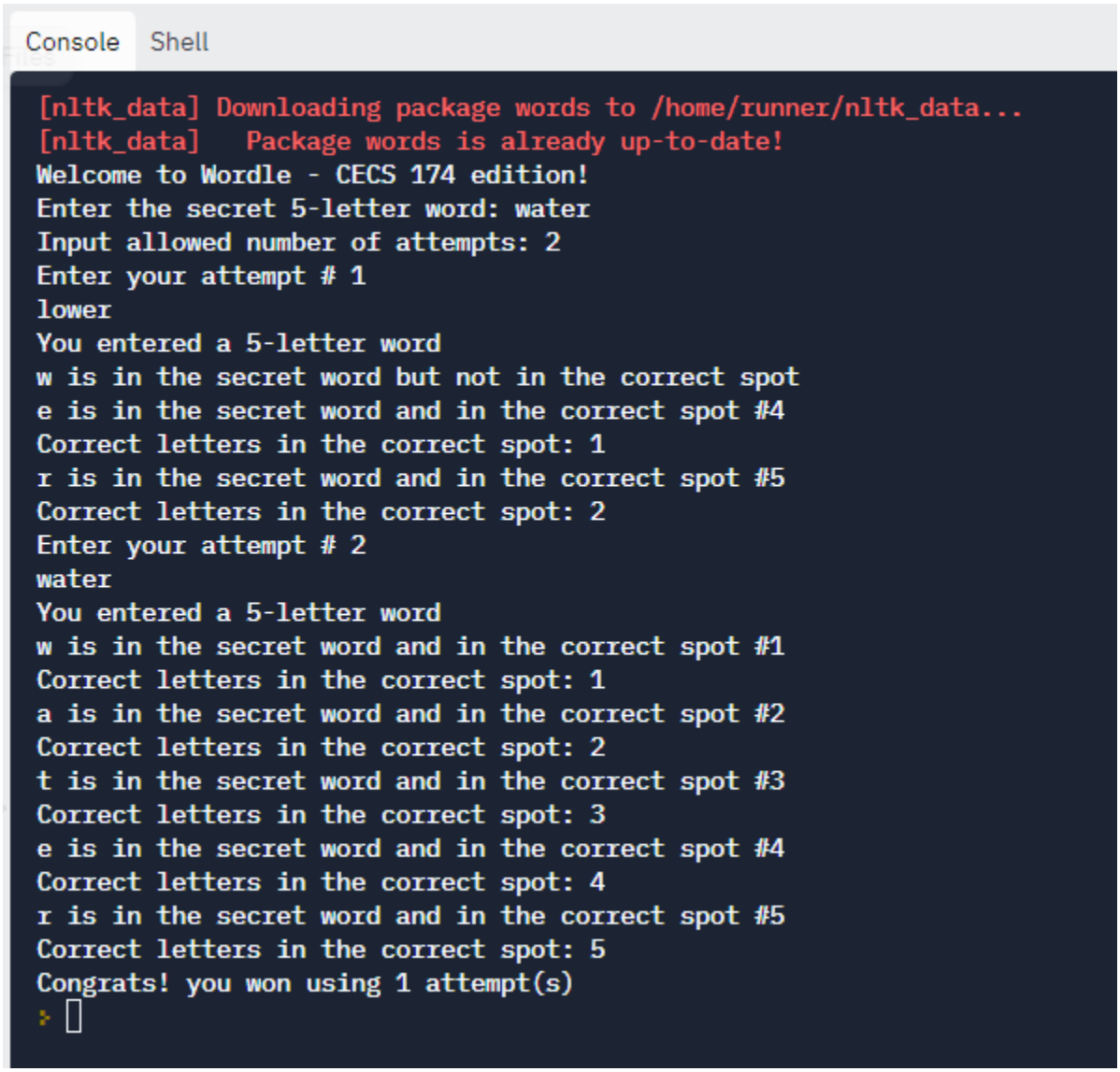
main.py

```
import nltk
nltk.download('words')
from nltk.corpus import words

print("Welcome to Wordle - CECS 174 edition!")
secret_word = input('Enter the secret 5-letter word: ')
count = len(secret_word)
is_word = False


while (count != 5) or (is_word == False):
  if secret_word in words.words():
    is_word = True

  else:
    is_word = False
  if (count != 5) or (is_word == False):
    print('Not a valid word, try again!')
    secret_word = input('Enter the secret 5-letter word: ')
    count = len(secret_word)

N = int(input("Input allowed number of attempts: "))
attempts = 0

while(attempts < N):

  print(f'Enter your attempt # {attempts + 1}')
  player_word = input()
  count2 = len(player_word)
  is_word2 = False
  while (count2 != 5) or (is_word2 == False):
    if player_word in words.words():
      is_word2 = True

    else:
      is_word2 = False

    if (count2 != 5) or (is_word2 == False):
      if (is_word2 == False):
        print('Not a valid word try again!')
      elif (count2 != 5):
        print(f'You entered a {len(player_word)}-letter word, but a 5-letter word is needed. Try Again.')
      print(f'Enter your attempts # {attempts + 1}')
      player_word = input()
      count2 = len(player_word)
  print('You entered a 5-letter word')

  letter_in_the_right_spot = 0
  for i in range(0, len(player_word)):
    for j in range(0, len(secret_word)):
      if (player_word[i] == secret_word[j]):
        if (i == j):
          print(f'{player_word[i]} is in the secret word and in the correct spot #{i + 1}')
          letter_in_the_right_spot += 1
          print(f'Correct letters in the correct spot: {letter_in_the_right_spot}')

        elif (i != j):
          print(f'{player_word[i]} is in the secret word but not in the correct spot')

      else:
        continue

  if secret_word == player_word:
    print(f'Congrats! you won using {attempts} attempt(s)')
    break
  attempts += 1
  tries = attempts

if (N == 0):
  player_word = 'No'
if secret_word != player_word:
  if (N == 0):
    pass
  elif (N <= attempts):
    print(f'You already used #{tries} attempts. Better luck tomorrow!')

```

Project Wordle Output for Bad Input:



Project Wordle Output for Winning:



Project Wordle Output for Losing: