Assignment 5

Enrique Saracho Felix
100406980
CPSC 1150
30/07/2023

Exercise 1

Program SecretPhrase

File name: SecretPhrase.java

Purpose: To allow the user to play a game in which they try to guess a random phrase in the least amount of tries possible, displaying the phrase's letters replaced by asterisks as a hint.

Packages: javax.swing.JOptionsPane
java.io.File
java.util.Scanner

Input: There are three possible arguments when calling the program, one for the number of rounds to play, one for selection if the phrases come from a file or a default set, and one for inserting the file name.

When playing the game, the inputs are characters.

Output: When calling the program without arguments, the command line displays the instructions for running it with arguments.

When playing, the program displays many dialog boxes with the state of the game. The box contains the number of round, instructions, and the target phrase covered by asterisks in the unguessed characters.‘

The results of the game are displayed in the command line in tabular form. One column for rounds, another for the target phrases, and one for the scores. The average score is displayed below the table.

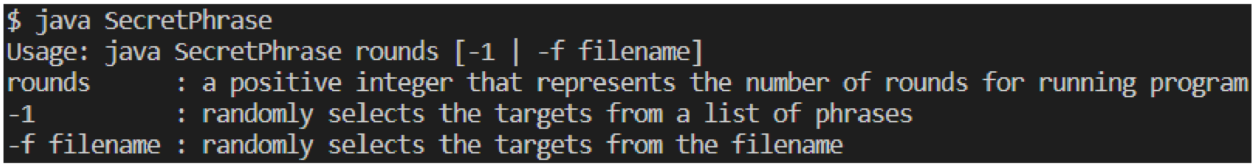
Pseudocode:

```
Algorithm SecretPhrase
START
        Set phrases as array of strings of size 100
        Set rounds as integer
        Set scores as array of floats of size 15
        Set guessed as array of floats of size 15

(main, parameter: args(array of strings))
        If ( length of args = 2 or 3 ) {
                rounds = args[0]
                If ( args[1] = -1 )
                        getPhrases()
        Else if ( args[1] = -f )
            getPhrases(args[2])
        Else {
            Print error message
            Exit program
        }

        Set sum as float = 0
        For ( i in range [0, rounds)) {
            playRound(i)
            sum += scores[i]
        }
        Set avg as float = sum / rounds

        printResults(avg)
    } Else {
        Print argument instructions
    }

(playRound, parameter round(integer))
    Set random as random integer in range [0, getPhrasesLength(phrases))
    Set ogPhrase as string = phrases[random]
    Set phrase as string = uppercase(ogPhrase)
    Set guesses as array of characters of size 50
    Set guess as integer = 0

    While ( replaceLetters(phrase, guesses) != phrase ) {
        guesses[guess] = getInput(replaceLetters(phrase, guesses), round)
        guess += 1
    }

    Set score as float = length of phrase(without spaces) / guess

    Print round, ogPhrase, score
    scores[round] = score
    guessed[round] = ogPhrase

(replaceLetters, parameters: phrase(string), guesses(array of characters))
    Set replacedPhrase as string = “”
    for ( i in range [0, length of phrase) ) {
        if ( phrase[ i ] == “ “ )
            replacedPhrase += “ “
        else if ( findCharacter(phrase[ i ], guesses) )
            replacedPhrase += phrase[ i ]
        else
            replacedPhrase += “*“
    }
    return replacedPhrase

(getInput, parameter: phrase(string), round(integer))
    Set guess as character
    Print phrase, round
    Read guess
    uppercase(guess)
    Return guess

(findCharacter, parameters: letter(character), guesses(array of characters))
    For ( i in range [0, length of guesses) ) {
        If ( guesses[ i ] = 0 )
            Break loop
        Else if ( guesses[ i ] = letter )
            Return true
    }
    Return false

(getPhrases)
    Set defaultPhrases as array of strings = {10 random phrases here}
    For ( i in range [0, length of defaultPhrases) )
        phrases[i] = defaultPhrases[i]

(getPhrases, parameter: fileName(string))
    Set file as file object of name fileName

    If ( can’t read file ) {
        Print error message
        Exit program
    }

    Set i as integer = 0
    While ( file has line of text ) {
        Set line as string = next line of text
        phrases[i] = line
        i += 1
    }

(getPhrasesLength, parameter phrases(array of strings))
    Set count as integer = 0
    Set i as integer = 0
    While ( phrases[i] != null ) {
        count += 1
        i += 1
    }
    Return count
```

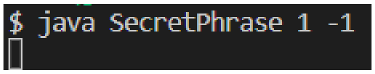
```
(printResults, parameter avg(float))
        Print header row
        For ( i in range [0, rounds) )
                Print i + 1, guessed[i], scores[i]
        Print avg
```

END *SecretPhrase*

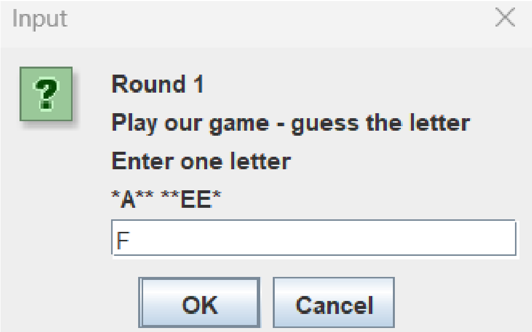
Test run(s):



```
$ java SecretPhrase
Usage: java SecretPhrase rounds [-1 | -f filename]
rounds      : a positive integer that represents the number of rounds for running program
-1          : randomly selects the targets from a list of phrases
-f filename : randomly selects the targets from the filename
```



```
$ java SecretPhrase 1 -1
```



Input

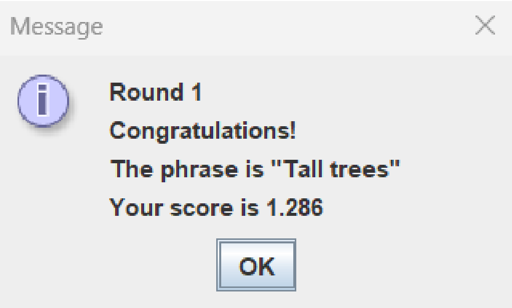
Round 1

Play our game - guess the letter

Enter one letter

*A** **EE*

F



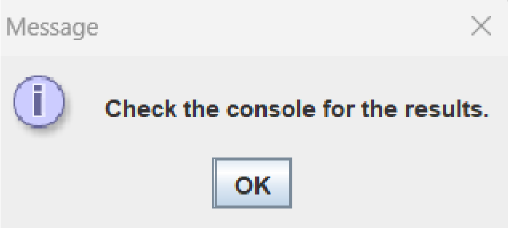
Message

Round 1

Congratulations!

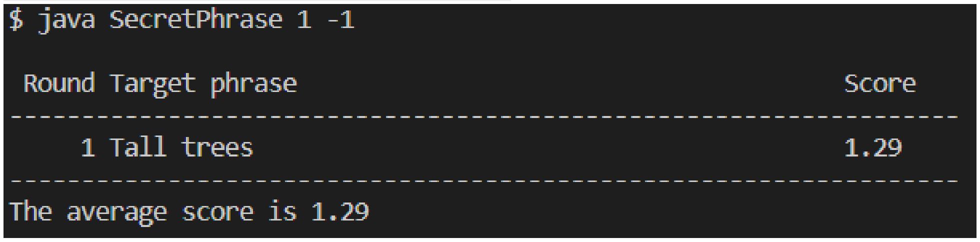
The phrase is "Tall trees"

Your score is 1.286

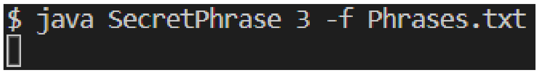


Message

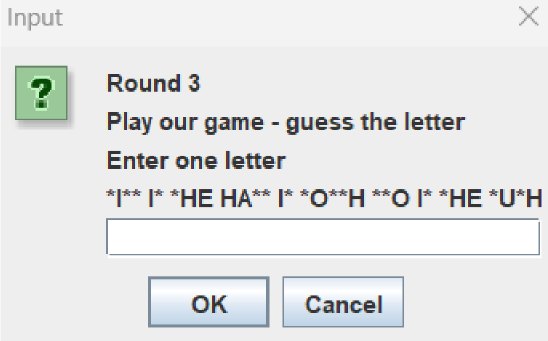
Check the console for the results.



```
$ java SecretPhrase 1 -1

 Round Target phrase                                          Score
------------------------------------------------------------------
     1 Tall trees                                             1.29
------------------------------------------------------------------
The average score is 1.29
```



```
$ java SecretPhrase 3 -f Phrases.txt
```

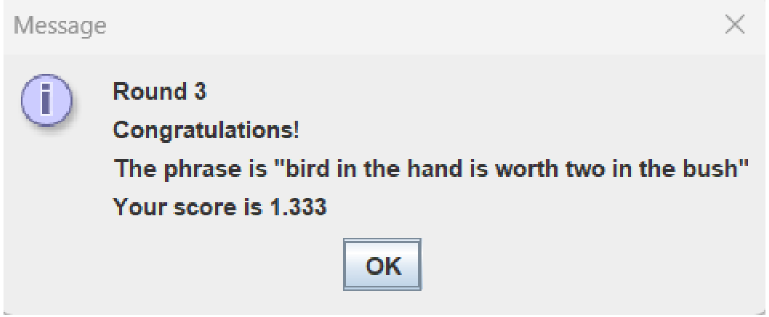


Round 3

Play our game - guess the letter

Enter one letter

*I** I* *HE HA** I* *O**H **O I* *HE *U*H

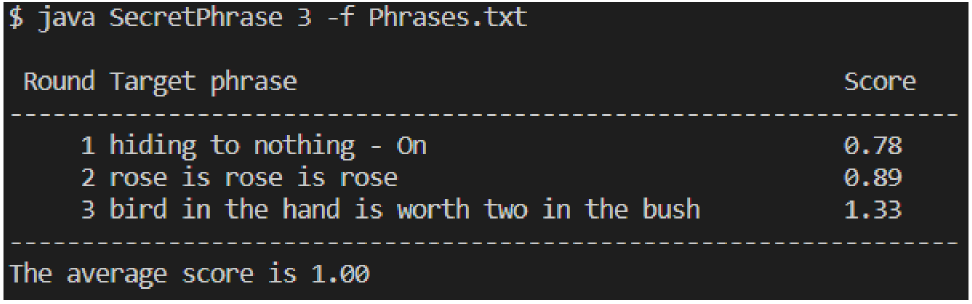


Round 3

Congratulations!

The phrase is "bird in the hand is worth two in the bush"

Your score is 1.333



```
$ java SecretPhrase 3 -f Phrases.txt

 Round Target phrase                                      Score
------------------------------------------------------------------
     1 hiding to nothing - On                             0.78
     2 rose is rose is rose                               0.89
     3 bird in the hand is worth two in the bush          1.33

------------------------------------------------------------------
The average score is 1.00
```