Bruce Dong

Oct.17.2020

CSDS 132

Professor. Connamacher

Project 2 Test Report

In this project, there are seven methods, and I will test each one of them and follow the Test 0,1,many first last middle principle

1. Method 1
2. Test 0, 1, and many:

> HW2.replaceFirstK("aaabc", 'a', 'd' ,0)

"aaabc"

> HW2.replaceFirstK("abc", 'a', 'd' ,1)

"dbc"

> HW2.replaceFirstK("aabc", 'a', 'd' ,2)

"ddbc"

> HW2.replaceFirstK("aaaaaabc", 'a', 'd' ,2)

"dddaaabc" // Test when input int is less then possible change letter

> HW2.replaceFirstK("aaaaaabc", 'a', 'd' ,60)

"ddddddbc" // Test when input int are more then possible change letter

1. Test first, last, and middle

> HW2.replaceFirstK("abc", 'a', 'd' ,1)

"dbc"

> HW2.replaceFirstK("abc", 'b', 'd' ,1)

"adc"

> HW2.replaceFirstK("abc", 'c', 'd' ,1)

"abd"

1. Method 2
2. Test first, last, and middle

> HW2.allChars('a','b')

"ab"

> HW2.allChars('a','z')

"abcdefghijklmnopqrstuvwxyz"

> HW2.allChars('y','z')

"yz"

> HW2.allChars('z','z')

"z"

1. Method 3
2. Test first, middle and last

> HW2.showCharOfString( "abc" , "a")

"a\_\_"

> HW2.showCharOfString( "abc" , "b")

"\_b\_"

> HW2.showCharOfString( "abc" , "c")

"\_\_c"

> HW2.showCharOfString( "abc" , "ac")

"a\_c"

> HW2.showCharOfString( "abc" , "ab")

"ab\_"

> HW2.showCharOfString( "abc" , "abc")

"abc"

> HW2.showCharOfString( "abcd" , "bc")

"\_bc\_"

1. Method 4

Since this method is different, I will copy everything in the interaction pane

1. Test 0,1 and many

> HW2.hangman("a", 0)

current word: \_ Bad guess times: 0

All of your guess letters are:

Correct Guess

All of your guess letters are: a

The word is: a and you win!

true

> HW2.hangman("a", 0)

current word: \_ Bad guess times: 0

All of your guess letters are:

Bad Guess

All of your guess letters are: b

The word is: a and you die!

false

> HW2.hangman("a", 1)

current word: \_ Bad guess times: 0

All of your guess letters are:

Correct Guess

All of your guess letters are: a

The word is: a and you win!

true

> HW2.hangman("a", 1)

current word: \_ Bad guess times: 0

All of your guess letters are:

Bad Guess

All of your guess letters are: b

The word is: a and you die!

false

> HW2.hangman("a", 2)

current word: \_ Bad guess times: 0

All of your guess letters are:

Correct Guess

All of your guess letters are: a

The word is: a and you win!

true

> HW2.hangman("a", 2)

current word: \_ Bad guess times: 0

All of your guess letters are:

Bad Guess

current word: \_ Bad guess times: 1

All of your guess letters are: b

Bad Guess

All of your guess letters are: bf

The word is: a and you die!

False

1. Test front back middle

> HW2.hangman("a", 2)

current word: \_ Bad guess times: 0

All of your guess letters are:

Correct Guess

All of your guess letters are: a

The word is: a and you win!

true

> b

Static Error: Undefined name 'b'

> HW2.hangman("abc", 2)

current word: \_\_\_ Bad guess times: 0

All of your guess letters are:

Correct Guess

current word: a\_\_ Bad guess times: 0

All of your guess letters are: a

Correct Guess

current word: ab\_ Bad guess times: 0

All of your guess letters are: ab

Correct Guess

All of your guess letters are: abc

The word is: abc and you win!

true

> HW2.hangman("abc", 2)

current word: \_\_\_ Bad guess times: 0

All of your guess letters are:

Correct Guess

current word: a\_\_ Bad guess times: 0

All of your guess letters are: a

Bad Guess

current word: a\_\_ Bad guess times: 1

All of your guess letters are: af

Bad Guess

All of your guess letters are: afg

The word is: abc and you die!

false

> HW2.hangman("abc", 2)

current word: \_\_\_ Bad guess times: 0

All of your guess letters are:

Correct Guess

current word: \_b\_ Bad guess times: 0

All of your guess letters are: b

Bad Guess

current word: \_b\_ Bad guess times: 1

All of your guess letters are: bf

Bad Guess

All of your guess letters are: bfg

The word is: abc and you die!

false

> HW2.hangman("abc", 2)

current word: \_\_\_ Bad guess times: 0

All of your guess letters are:

Correct Guess

current word: \_\_c Bad guess times: 0

All of your guess letters are: c

Bad Guess

current word: \_\_c Bad guess times: 1

All of your guess letters are: cf

Bad Guess

All of your guess letters are: cfg

The word is: abc and you die!

False

1. Special situation: type in same letter will make no effect

In this situation, I typed two times of a and f, and nothing changes

> HW2.hangman("abc", 2)

current word: \_\_\_ Bad guess times: 0

All of your guess letters are:

Correct Guess

current word: a\_\_ Bad guess times: 0

All of your guess letters are: a

Correct Guess

You should type a different letter!

current word: a\_\_ Bad guess times: 0

All of your guess letters are: a

Bad Guess

current word: a\_\_ Bad guess times: 1

All of your guess letters are: af

You should type a different letter!

current word: a\_\_ Bad guess times: 1

All of your guess letters are: af

Correct Guess

current word: ab\_ Bad guess times: 1

All of your guess letters are: afb

Correct Guess

All of your guess letters are: afbc

The word is: abc and you win!

True

1. Method 5
2. Test first back middle

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','a'}, "a")

true

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','a'}, "br")

true

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','a'}, "abr")

true

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','a'}, "ada")

true

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','x'}, "x")

true

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','x'}, "dx")

true

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','x'}, "racad")

True

1. Test 0, 1, many

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','x'}, "")

true

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','x'}, "a")

true

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','x'}, "rac")

True

1. Special case: forward and backward search

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','x'}, "rac")

true

> HW2.hiddenString(new char[]{'a','b','r','a','c','a','d','x'}, "car")

True

1. Method 6

Since this method has too much conditions, Test 0 and Test 1 will be within Test horizontal condition. Also, since most of them are in “many” condition, it is meaningless to repeat it.

1. Horizontal condition Test front back middle and 0 1 many forward and backwards

> HW2.hiddenString(new char[][]{{'a', 'b','a', 'c'},{'e','r','c','a','d'},{'b','r'}}, "ace")

false

> HW2.hiddenString(new char[][]{{'a', 'b','a', 'c'},{'e','r','c','a','d'},{'b','r'}}, "a")

true

> HW2.hiddenString(new char[][]{{'a', 'b','a', 'c'},{'e','r','c','a','d'},{'b','r'}}, "c")

true

> HW2.hiddenString(new char[][]{{'a', 'b','a', 'c'},{'e','r','c','a','d'},{'b','r'}}, "r")

true

> HW2.hiddenString(new char[][]{{'a', 'b','a', 'c'},{'e','r','c','a','d'},{'b','x'}}, "x")

true

> HW2.hiddenString(new char[][]{{'a', 'b','a', 'c'},{'e','r','c','a','d'},{'b','x'}}, "erc")

true

> HW2.hiddenString(new char[][]{{'a', 'b','a', 'c'},{'e','r','c','a','d'},{'b','x'}}, "ace")

false

> HW2.hiddenString(new char[][]{{'a', 'b','a', 'c'},{'e','r','c','a','d'},{'b','x'}}, "aba")

true

> HW2.hiddenString(new char[][]{{'a', 'b','a', 'c'},{'e','r','c','a','d'},{'b','x'}}, "bx")

True

> HW2.hiddenString(new char[][]{{'a', 'b','a', 'c'},{'e','r','c','a','d'},{'b','x'}}, "xb")

true

1. Vertical Condition: front middle back forward and backwards

This array looks like:

{a,f,j,n,q,s}

{b,g,k,o,r}

{c,h,I}

{d,i,m,p}

{e,}

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "abcd")

True // Test front line

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "dcba")

True // Test front line back wards

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "dcb")

True // Test middle line

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "fghi")

true// Test middle line back wards

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "ghi")

True // Test middle line

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "qr")

True // Test last line

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "rq")

True // Test last line back wards

1. Test diagonal line

This array looks like:

{a,f,j,n,q,s}

{b,g,k,o,r}

{c,h,I}

{d,i,m,p}

{e,}

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "aglp")

True // Test diagonal line from left corner to right forward

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "plga")

True // Test diagonal line from left corner to right backward

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "gl")

True // Test diagonal line in middle

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "eiloq")

True // Test diagonal line from right corner to left forward

> HW2.hiddenString(new char[][]{{'a', 'f', 'j','n','q','s'},{'b','g','k','o','r'},{'c','h','l'},{'d','i','m','p'},{'e'}}, "qolie")

True // Test diagonal line from right corner to left backward

1. Method 7
2. Test 0,1,many

> HW2.capitalizeWords("")

""

> HW2.capitalizeWords("a")

"a"

> HW2.capitalizeWords("A")

"A"

> HW2.capitalizeWords("Abc")

"ABC"

> HW2.capitalizeWords("abc")

"abc"

1. Test front, back and middle

> HW2.capitalizeWords("abc Abc")

"abc ABC"

> HW2.capitalizeWords("Abc abc")

"ABC abc"

> HW2.capitalizeWords("abc Abc aBc")

"abc ABC ABC"

1. Special condition: when we meet punctuation and ‘-‘ and multiple ‘ ‘

> HW2.capitalizeWords("abc? Abc?? abc ? abc-abc /// Abc-abc // abc-aBc?")

"abc? ABC?? abc ? abc-abc /// ABC-ABC // ABC-ABC?"

Until now, all 7 methods have been tested and all of them work properly.