Author: Christopher Tam

**matchingParentheses method:**

For this method, the ‘test 0, test 1, test many’ approach will consider string inputs with 0, 1, and more than 1 matching/unmatching parentheses. The ‘test first, middle, last’ approach will consider string inputs containing matching and nonmatching parentheses at the start, the end, and the middle of the string.

**Matching parentheses**

Test 0

1. For the given case, the expected output should be false.

> HW2.matchingParentheses("This )is a (test of the( (matching parentheses")

false

Test 1

1. For the given case, the expected output should be true.

> HW2.matchingParentheses("This is a (test of the) matching parentheses")

true

Test many

1. For the given case, the expected output should be true.

> HW2.matchingParentheses("This is a (test (of) the) (matching) parentheses")

true

Test first

1. For the given case, the expected output should be true.

> HW2.matchingParentheses("(This is a) test of the matching parentheses")

true

Test middle

1. For the given case, the expected output should be true.

> HW2.matchingParentheses("This is a test (of the) matching parentheses")

True

Test last

1. For the given case, the expected output should be true.

> HW2.matchingParentheses("This is a test of the matching (parentheses)")

true

**Unmatching Parentheses**

Test 0

1. For the given case, the expected output should be true.

> HW2.matchingParentheses("This is a (test (of) the) (matching) parentheses")

true

Test 1

1. For the given case, the expected output should be false.

> HW2.matchingParentheses("This ) is a (test (of) the) (matching) parentheses")

false

Test many

1. For the given case, the expected output should be false.

> HW2.matchingParentheses("This ) is) a (test (of) the) (matching) (parentheses")

false

Test first

1. For the given case, the expected output should be false.

> HW2.matchingParentheses(")This (is a test of the matching parentheses")

false

Test middle

1. For the given case, the expected output should be false.

> HW2.matchingParentheses("This is a )test (of the matching parentheses")

false

Test last

1. For the given case, the expected output should be false.

> HW2.matchingParentheses("This is a test of the matching )parentheses(")

false

**everyNthExcept method:**

For this method, the ‘test 0, test 1, test many’ approach will consider string inputs with length of 0, 1, and more than 1, and for each of these cases each of the integer inputs with values of 0, 1, and more than 1 (for each of the integer input values) will be considered. The ‘test first, middle, last’ approach is not needed for this method.

**String input**

Test 0

**Int start input**

Test 0

1. For the given case, the expected output is an error message.

> String s = ""

> HW2.everyNthExcept(s,0,5,6)

java.lang.StringIndexOutOfBoundsException: String index out of range: -1

at java.lang.String.charAt(Unknown Source)

at HW2.everyNthExcept(HW2.java:31)

Test 1

1. For the given case, the expected output is “”.

> String s = ""

> HW2.everyNthExcept(s,1,5,6)

""

Test many

1. For the given case, the expected output is “”.

> String s = ""

> HW2.everyNthExcept(s,3,5,6)

""

**Int skip input**

Test 0

1. For the given case, the expected output is “”.

> String s = ""

> HW2.everyNthExcept(s,2,0,6)

""

Test 1

1. For the given case, the expected output is “”.

> String s = ""

> HW2.everyNthExcept(s,2,1,6)

""

Test many

1. For the given case, the expected output is “”.

> String s = ""

> HW2.everyNthExcept(s,2,1,6)

""

**Int except input**

Test 0

1. For the given case, the expected output is “”.

> String s = ""

> HW2.everyNthExcept(s,2,4,0)

""

Test 1

1. For the given case, the expected output is “”.

> String s = ""

> HW2.everyNthExcept(s,2,4,1)

""

Test many

1. For the given case, the expected output is “”.

> String s = ""

> HW2.everyNthExcept(s,2,4,5)

""

**String input**

Test 1

**Int start input**

Test 0

1. For the given case, the expected output is an error message.

> String s = "a"

> HW2.everyNthExcept(s,0,4,5)

java.lang.StringIndexOutOfBoundsException: String index out of range: -1

at java.lang.String.charAt(Unknown Source)

at HW2.everyNthExcept(HW2.java:31)

Test 1

1. For the given case, the expected output is “a”.

> String s = "a"

> HW2.everyNthExcept(s,1,4,5)

"a"

Test many

1. For the given case, the expected output is “”.

> String s = "a"

> HW2.everyNthExcept(s,3,4,5)

""

**Int skip input**

Test 0

1. For the given case, the expected output is “”.

> String s = "a"

> HW2.everyNthExcept(s,2,0,5)

""

Test 1

1. For the given case, the expected output is “”.

> String s = "a"

> HW2.everyNthExcept(s,2,1,5)

""

Test many

1. For the given case, the expected output is “”.

> String s = "a"

> HW2.everyNthExcept(s,2,3,5)

""

**Int except input**

Test 0

1. For the given case, the expected output is “”.

> String s = "a"

> HW2.everyNthExcept(s,2,3,0)

""

Test 1

1. For the given case, the expected output is “”.

> String s = "a"

> HW2.everyNthExcept(s,2,3,1)

""

Test many

1. For the given case, the expected output is “”.

> String s = "a"

> HW2.everyNthExcept(s,2,3,4)

""

**String input**

Test many

**Int start input**

Test 0

1. For the given case, the expected output is an error message.

> String s = "Thankyou for this project Harold"

> HW2.everyNthExcept(s,0,3,4)

java.lang.StringIndexOutOfBoundsException: String index out of range: -1

at java.lang.String.charAt(Unknown Source)

at HW2.everyNthExcept(HW2.java:31)

at sun.reflect.GeneratedMethodAccessor2.invoke(Unknown Source)

Test 1

1. For the given case, the expected output is “Tnofipjal”.

> String s = "Thankyou for this project Harold"

> HW2.everyNthExcept(s,1,3,4)

"Tnofipjal"

Test many

1. For the given case, the expected output is “ay r oco”.

> String s = "Thankyou for this project Harold"

> HW2.everyNthExcept(s,3,3,4)

"ay r oco"

**Int skip input**

Test 0

1. For the given case, the expected output is an error message.

> String s = "Thankyou for this project Harold"

> HW2.everyNthExcept(s,2,0,4)

java.lang.ArithmeticException: / by zero

at HW2.everyNthExcept(HW2.java:30)

at sun.reflect.GeneratedMethodAccessor2.invoke(Unknown Source)

Test 1

1. For the given case, the expected output is “Hankou or hisproectHarld”.

> String s = "Thankyou for this project Harold"

> HW2.everyNthExcept(s,2,1,4)

"hankou or hisproectHarld"

Test many

1. For the given case, the expected output is “h”.

> String s = "Thankyou for this project Harold"

> HW2.everyNthExcept(s,2,4,4)

"h"

**Int except input**

Test 0

1. For the given case, the expected output is an error message.

> String s = "Thankyou for this project Harold"

> HW2.everyNthExcept(s,2,4,0)

java.lang.ArithmeticException: / by zero

at HW2.everyNthExcept(HW2.java:34)

at sun.reflect.GeneratedMethodAccessor2.invoke(Unknown Source)

Test 1

1. For the given case, the expected output is “h”.

> String s = "Thankyou for this project Harold"

> HW2.everyNthExcept(s,2,4,1)

"h"

Test many

1. For the given case, the expected output is “hyft o”.

> String s = "Thankyou for this project Harold"

> HW2.everyNthExcept(s,2,4,5)

"hyft o"

**flipEachK method:**

For this method, the ‘test 0, test 1, test many’ approach will consider string inputs with length of 0, 1, and more than 1, and for each of these cases integer input with a value of 0, 1, and more than 1 (for each of the integer input values) will be considered. The ‘test first, middle, last’ approach will consider k reversal points at the middle and end points of the string. It is not possible to reverse the first character of the string.

**String Input**

Test 0

**Int k input**

Test 0

1. For the given case, the expected output is “”.

> String s = ""

> HW2.flipEachK(s,0)

""

Test 1

1. For the given case, the expected output is “”.

> String s = ""

> HW2.flipEachK(s,1)

""

Test many

1. For the given case, the expected output is “”.

> String s = ""

> HW2.flipEachK(s,5)

""

**String Input**

Test 1

**Int k input**

Test 0

1. For the given case, the expected output is an error message.

> String s = "a"

> HW2.flipEachK(s,0)

java.lang.ArithmeticException: / by zero

at HW2.flipEachK(HW2.java:46)

Test 1

1. For the given case, the expected output is “a”.

> String s = "a"

> HW2.flipEachK(s,1)

"a"

Test many

1. For the given case, the expected output is “a”.

> String s = "a"

> HW2.flipEachK(s,4)

"a"

**String Input**

Test many

**Int k input**

Test 0

1. For the given case, the expected output is an error message.

> String s = "adfasfdasd"

> HW2.flipEachK(s,0)

java.lang.ArithmeticException: / by zero

at HW2.flipEachK(HW2.java:46)

Test 1

1. For the given case, the expected output is “adfasfdasd”.

> String s = "adfasfdasd"

> HW2.flipEachK(s,1)

"adfasfdasd"

Test many

1. For the given case, the expected output is “adffsadasd”.

> String s = "adfasfdasd"

> HW2.flipEachK(s,3)

"adffsadasd"

**String Input**

Test middle

1. For the given case, the expected output is “asdffdsaasd”.

> String s = "asdfasdfasd"

> HW2.flipEachK(s,4)

"asdffdsaasd"

Test last

1. For the given case, the expected output is “asdffdsaasdfdsa”.

> String s = "asdfasdfasdfasd"

> HW2.flipEachK(s,4)

"asdffdsaasdfdsa"

**reverseDigits method:**

For this method, the ‘test 0, test 1, test many’ approach will consider string inputs with length of 0, 1, and more than 1, and the ‘test first, middle, last’ approach will consider strings with reversible digits at first, middle, and last characters of the string.

Test 0

1. For the given case, the expected output is “”.

> String a = ""

> HW2.reverseDigits(a)

""

Test 1

1. For the given case, the expected output is “7”.

> String a = "7"

> HW2.reverseDigits(a)

"7"

Test many

1. For the given case, the expected output is “8 asdf 4 9 asdf4 fas5oo7pp”.

> String a = "7 asdf 5 4 asdf9 fas4oo8pp"

> HW2.reverseDigits(a)

"8 asdf 4 9 asdf4 fas5oo7pp"

Test first

1. For the given case, the expected output is “8 asdf 4 9 asdf4 fas5oo7pp”.

> String a = "7 asdf 5 4 asdf9 fas4oo8pp"

> HW2.reverseDigits(a)

"8 asdf 4 9 asdf4 fas5oo7pp"

Test middle

1. For the given case, the expected output is “asdf 8 4 asdf9 fas4oo5pp”.

> String a = "asdf 5 4 asdf9 fas4oo8pp"

> HW2.reverseDigits(a)

"asdf 8 4 asdf9 fas4oo5pp"

Test last

1. For the given case, the expected output is “asdf 3 8 asdf4 fas9oo4pp5”.

> String a = "asdf 5 4 asdf9 fas4oo8pp3"

> HW2.reverseDigits(a)

"asdf 3 8 asdf4 fas9oo4pp5"

**editOut method:**

For this method, the ‘test 0, test 1, test many’ approach will consider string inputs with length of 0, 1, and more than 1, and the ‘test first, middle, last’ approach will consider strings with removed and retained matched parentheses at the start, middle, and end of the string.

Test 0

1. For the given case, the expected output is “”.

> String a = ""

> HW2.editOut(a)

""

Test 1

1. For the given case, the expected output is “Input does not contain matching parentheses!”

> String a = "("

> HW2.editOut(a)

"Input does not contain matching parentheses!"

Test many

1. For the given case, the expected output is “adsfasdasdf asdf asdf”.

> String a = "adsfasd((asdf)) (asdf(asdf)) asdf (l)"

> HW2.editOut(a)

"adsfasdasdf asdf asdf "

**Removed parentheses:**

Test first

1. For the given case, the expected output is “ asdf asdf”.

> String a = "(((adsfasd)))(asdf)(asdf(asdf)) asdf"

> HW2.editOut(a)

" asdf asdf"

Test middle

1. For the given case, the expected output is “ads asdf”.

> String a = "ads(((asdf)))(asdf(asdf))"

> HW2.editOut(a)

"ads asdf"

Test last

1. For the given case, the expected output is "adsasdf ".

> String a = "ads((asdf))((asdf(asdf)))"

> HW2.editOut(a)

"adsasdf "

**Retained parentheses:**

Test first

1. For the given case, the expected output is “adsasdfdf”.

> String a = "((ads))((asdf))df"

> HW2.editOut(a)

"adsasdfdf"

Test middle

1. For the given case, the expected output is “adsasdfdf”.

> String a = "ads((asdf))df"

> HW2.editOut(a)

"adsasdfdf"

Test last

1. For the given case, the expected output is “adsasdfdf”.

> String a = "adsasdf((df))"

> HW2.editOut(a)

"adsasdfdf"

**replaceText method:**

For this method, the ‘test 0, test 1, test many’ approach will consider string inputs with length of 0, 1, and more than 1 for both string inputs, and the ‘test first, middle, last’ approach will consider replaced matched parentheses at the start, middle, and end of the first input string.

**String input string1:**

Test 0

1. For a string input of zero length, the expected output is "Input does not contain matching parentheses!"

Test 1

1. For a string input of length 1, the expected output should be the string itself.

Test many

1. For a string input of length >1, the first substring should be replaced by its corresponding substring in the second input string, and so on.

**String input string2:**

Test 0

1. For a string input of length 0, the first string input should be returned.

Test 1

1. For a string input of length 1, the first string input should be returned.

Test many

1. For a string input of length > 1, the first substring should be returned with its corresponding substrings replaced accordingly.

Test first

1. For a string input with matching parentheses at the start of the string, this substring should be replaced by its corresponding substring in the second input string.

Test middle

1. For a string input with matching parentheses at the start of the string, this substring should be replaced by its corresponding substring in the second input string.

Test last

1. For a string input with matching parentheses at the start of the string, this substring should be replaced by its corresponding substring in the second input string.