**6 kyu**

**Braces status**

11384% of 113122 of270[bestwebua](https://www.codewars.com/users/bestwebua)

Python

* [TRAIN AGAIN](https://www.codewars.com/kata/braces-status/train/python)
* [NEXT KATA](https://www.codewars.com/trainer/python)

Details

[Solutions](https://www.codewars.com/kata/braces-status/solutions/python)

[Forks (1)](https://www.codewars.com/kata/braces-status/forks/python)

[Discourse (24)](https://www.codewars.com/kata/braces-status/discuss/python)

* Add to Collection
* |
* Share this kata:

Write a function that checks the braces status in a string, and return True if all braces are properly closed, or False otherwise. Available types of brackets: (), [], {}.

**Please note, you need to write this function without using regex!**

Examples

'([[some](){text}here]...)' => True

'{([])}' => True

'()[]{}()' => True

'(...[]...{(..())}[abc]())' => True

'1239(df){' => False

'[()])' => False

')12[x]34(' => False

Don't forget to rate this kata! Thanks :)

<https://www.codewars.com/kata/braces-status/python>

*# Python program for linked list implementation of stack*

*# Class to represent a node*

**class** StackNode:

*# Constructor to initialize a node*

**def** \_\_init\_\_(self, data):

        self.data = data

        self.next = None

**class** Stack:

*# Constructor to initialize the root of linked list*

**def** \_\_init\_\_(self):

        self.root = None

**def** isEmpty(self):

**return** True **if** self.root **is** None **else**  False

**def** push(self, data):

        newNode = StackNode(data)

        newNode.next = self.root

        self.root = newNode

*#print "%d pushed to stack" %(data)*

**def** pop(self):

**if** (self.isEmpty()):

**return** float("-inf")

        temp = self.root

        self.root = self.root.next

        popped = temp.data

**return** popped

**def** peek(self):

**if** self.isEmpty():

**return** float("-inf")

**return** self.root.data

**def** mismoTipo(abierto):

**if**(abierto == '('): **return** ')'

**if**(abierto == '['): **return** ']'

**if**(abierto == '{'): **return** '}'

**return** 'x'

**def** braces\_status(s):

    pila = Stack()

    abiertos = '([{'

    cerrados = ')]}'

**for** i **in** range(0, len(s)) :

**if**(s[i] **in** abiertos):

            pila.push(s[i])

*#if (pila.isEmpty()):return False*

**if**(s[i] **in** cerrados):

**if**(mismoTipo( pila.peek()) == s[i] ):

                pila.pop()

**else**:

**return** False

**return** pila.isEmpty()

*#s = "{}{)"*

s = '[][)[][]'

**print**(braces\_status(s))

s = '(12[x]34)'

**print**(braces\_status(s))

s = "()y(){}y{(){}()}" *#True*

**print**(braces\_status(s))

*#expecting: "True"*