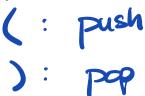
CSC148 - Balancing Parentheses

We are writing client code and need a function (outside the class) to determine whether the parentheses in an expression are balanced: opening and closing parentheses match and are properly nested inside each other.

]	For four examples, we'll give you a string one character at a time. Your job is to determine whether the string has balanced parentheses or not. <i>Don't just write down every character without thinking!</i> Instead, use a stack to keep track of the minimum amount of information you need to solve the problem.			
	Expression 1:		Expression 2:	
	Expression 1.		Expression 2.	
ı	1		I I	
L	Stack		Stack	
	Were the parentheses balanced?		Were the parentheses balan	ced?
	Yes No		Yes No	ccu:
	Expression 3:		Expression 4:	
L				
	Stack		Stack	
	Were the parentheses balanced?		Were the parentheses balan	ced?
	Yes No		Yes No	

- 2. We need a general strategy that will work in all cases. To find it, answer these questions:
 - (a) What will you do with each character as you receive it?



aufling else: ignore

See "vough work" for more

(b) At the end, how will you know whether the parentheses were balanced?

3. Now implement the function.

```
def is_balanced(line: str) -> bool:
    """Return whether <line> contains balanced parentheses.

Ignore square and curly brackets.

>>> is_balanced('(a * (3 + b))')
    True
    >>> is_balanced('(a * (3 + b)]') # Note that the two ']'s don't matter.
False
    >>> is_balanced('1 + 2(x - y))') # Note that the ']' doesn't matter.
    True
    >>> is_balanced('3 - (x'))
False
    """
```