'''

Problem:

Given two strings str1 and str2 and below operations that can performed on str1.

Find minimum number of edits (operations) required to convert ‘str1’ into ‘str2’.

The available operations include: insert, remove, replace

Ways:

The recursion method is a naive solution.

If the last character is the same, no operation;

else, try 3 available operations (insert, remove, replace) and select the minimal steps

Notice, the stop condition should be carefully disposed

Option 1:

l\_str1 = len(str1)

l\_str2 = len(str2)

if l\_str1 + l\_str2 == 0:

return 0

if (l\_str1 + l\_str2 > 0) and (l\_str1 \* l\_str2 == 0):

return 10000

Option 2:

if len(str1) == 0:

return len(str2)

if len(str2) == 0:

return len(str1)

Option 2 is more elegant

'''

def edit\_distance(str1, str2):

if len(str1) == 0:

return len(str2)

if len(str2) == 0:

return len(str1)

if str1[-1] == str2[-1]:

return edit\_distance(str1[:-1], str2[:-1])

else:

d\_insert = edit\_distance(str1, str2[:-1])

d\_remove = edit\_distance(str1[:-1], str2)

d\_replace = edit\_distance(str1[:-1], str2[:-1])

return 1 + min(d\_insert, d\_remove, d\_replace)

def edit\_distance\_optim(str1, str2):

m = len(str1)

n = len(str2)

arr = [[0] \* (n + 1) for i in range (m+1)]

for irow in range(m+1):

for jcol in range(n+1):

if irow == 0:

arr[irow][jcol] = jcol

if jcol == 0:

arr[irow][jcol] = irow

if str1[irow] == str2[jcol]:

arr[irow][jcol] = arr[irow-1][jcol-1]

else:

d\_insert = arr[irow][jcol-1]

d\_remove = arr[irow-1][jcol]

d\_replace = arr[irow-1][jcol-1]

value = min(d\_insert, d\_remove, d\_replace)

arr[irow][jcol] = value + 1

return arr, arr[m][n]

str1 = "sunday"

str2 = "saturday"

# min\_operation = edit\_distance(str1, str2)

# print(min\_operation)

arr, min\_operation = edit\_diatance(str1, str2)

print(arr)

print(min\_operation)