Reverse words in a given String in Python

Input : str = "geeks quiz practice code"

Output : str = "code practice quiz geeks"

# Function to reverse words of string

def reverseWords(input):

    # split words of string separated by space

    inputWords = input.split(" ")

    inputWords=inputWords[-1::-1]

    # now join words with space

    output = ' '.join(inputWords)

    return output

if \_\_name\_\_ == "\_\_main\_\_":

    input = 'geeks quiz practice code'

    print reverseWords(input)

# Ways to remove i’th character from string in Python

# Python code to demonstrate

# method to remove i'th character

# using slice + concatenation

# Initializing String

test\_str = "GeeksForGeeks"

# Printing original string

print ("The original string is : " + test\_str)

# Removing char at pos 3

# using slice + concatenation

new\_str = test\_str[:2] +  test\_str[3:]

# Printing string after removal

# removes ele. at 3rd index

print ("The string after removal of i'th character : " + new\_str)

# Python | Check if a Substring is Present in a Given String

Input : s1 = geeks s2=geeks for geeks

Output : yes

Input : s1 = geek s2=geeks for geeks

Output : yes

# function to check if small string is

# there in big string

def check(string, sub\_str):

    if (string.find(sub\_str) == -1):

        print("NO")

    else:

        print("YES")

# driver code

string = "geeks for geeks"

sub\_str ="geek"

check(string, sub\_str)

# Python | Program to accept the strings which contains all vowels

**Input :** geeksforgeeks

**Output :** Not Accepted

**Input :** ABeeIghiObhkUul

**Output :** Accepted

|  |
| --- |
| # Python program to accept the strings  # which contains all the vowels    # Function for check if string  # is accepted or not  def check(string) :        # set() function convert "aeiou"      # string into set of characters      # i.e.vowels = {'a', 'e', 'i', 'o', 'u'}      vowels = set("aeiou")        # set() function convert empty      # dictionary into empty set      s = set({})        # looping through each      # character of the string      for char in string :        # Check for the character is present inside      # the vowels set or not. If present, then      # add into the set s by using add method          if char in vowels :              s.add(char)          else:              pass        # check the length of set s equal to length      # of vowels set or not. If equal, string is      # accepted otherwise not      if len(s) == len(vowels) :          print("Accepted")      else :          print("Not Accepted")      # Driver code  if \_\_name\_\_ == "\_\_main\_\_" :        string = "SEEquoiaL"        string = string.lower()        # calling function      check(string) |
|  |

**Output:**

Accepted

# Python | Count the Number of matching characters in a pair of string

**Input :** str1 = 'abcdef'

str2 = 'defghia'

**Output :** 4

(i.e. matching characters :- a, d, e, f)

***Note:****For this, use****string.find(character)****in python.*

*This returns the first occurrence index of character in string, if found, otherwise return -1.*

*For example : str=’abcdedde’  
str.find(‘d’) –> 3  
str.find(‘e’) –> 4  
str.find(‘g’) –> -1*

|  |
| --- |
| # Python code to count number of matching  # characters in a pair of strings    # count function  def count(str1, str2):      c, j = 0, 0        # loop executes till length of str1 and      # stores value of str1 character by character      # and stores in i at each iteration.      for i in str1:            # this will check if character extracted from          # str1 is present in str2 or not(str2.find(i)          # return -1 if not found otherwise return the          # starting occurrence index of that character          # in str2) and j == str1.find(i) is used to          # avoid the counting of the duplicate characters          # present in str1 found in str2          if str2.find(i)>= 0 and j == str1.find(i):              c += 1          j += 1      print ('No. of matching characters are : ', c)    # Main function  def main():      str1 ='aabcddekll12@' # first string      str2 ='bb2211@55k' # second string      count(str1, str2) # calling count function    # Driver Code  if \_\_name\_\_=="\_\_main\_\_":      main() |

**Output :**

No. of matching characters are : 5

# Python code to count number of unique matching

# characters in a pair of strings

# count function count the common unique

# characters present in both strings .

def count(str1 ,str2) :

    # set of characters of string1

    set\_string1 = set(str1)

    # set of characters of string2

    set\_string2 = set(str2)

    # using (&) intersection mathematical operation on sets

    # the unique characters present in both the strings

    # are stored in matched\_characters set variable

    matched\_characters = set\_string1 & set\_string2

    # printing the length of matched\_characters set

    # gives the no. of matched characters

    print("No. of matching characters are : " + str(len(matched\_characters)) )

# Driver code

if \_\_name\_\_ == "\_\_main\_\_" :

    str1 = 'aabcddekll12@'  # first string

    str2 = 'bb2211@55k'     # second string

    # call count function

    count( str1 , str2 )

# Remove all duplicates from a given string in Python

Input : geeksforgeeks

Output : efgkos

from collections import OrderedDict

# Function to remove all duplicates from string

# and order does not matter

def removeDupWithoutOrder(str):

    # set() --> A Set is an unordered collection

    #            data type that is iterable, mutable,

    #            and has no duplicate elements.

    # "".join() --> It joins two adjacent elements in

    #               iterable with any symbol defined in

    #               "" ( double quotes ) and returns a

    #               single string

    return "".join(set(str))

# Function to remove all duplicates from string

# and keep the order of characters same

def removeDupWithOrder(str):

    return "".join(OrderedDict.fromkeys(str))

# Driver program

if \_\_name\_\_ == "\_\_main\_\_":

    str = "geeksforgeeks"

    print "Without Order = ",removeDupWithoutOrder(str)

    print "With Order = ",removeDupWithOrder(str)

# Program to check if a string contains any special character

**Examples :**

Input : Geeks$For$Geeks

Output : String is not accepted.

Input : Geeks For Geeks

Output : String is accepted

# Python program to check if a string

# contains any special character

# import required package

import re

# Function checks if the string

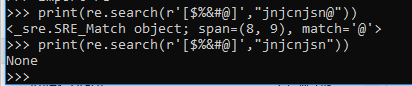
# contains any special character

def run(string):

    # Make own character set and pass

    # this as argument in compile method

    regex = re.compile('[@\_!#$%^&\*()<>?/\|}{~:]')



    # Pass the string in search

    # method of regex object.

    if(regex.search(string) == None):

        print("String is accepted")

    else:

        print("String is not accepted.")

# Driver Code

if \_\_name\_\_ == '\_\_main\_\_' :

    # Enter the string

    string = "Geeks$For$Geeks"

    # calling run function

    run(string)

# String slicing in Python to check if a string can become empty by recursive deletion

Input : str = "GEEGEEKSKS", sub\_str = "GEEKS"

Output : Yes

Explanation : In the string GEEGEEKSKS, we can first

delete the substring GEEKS from position 4.

The new string now becomes GEEKS. We can

again delete sub-string GEEKS from position 1.

Now the string becomes empty.

# Function to check if a string can become empty

# by recursively deleting a given sub-string

def checkEmpty(input, pattern):

    if len(input)== 0:

         return 'false'

    while (len(input) != 0):

          # find sub-string in main string

          index = input.find(pattern)

          # check if sub-string founded or not

          if (index ==(-1)):

              return 'false'

          # slice input string in two parts and concatenate

          input = input[0:index] + input[index + len(pattern):]

     return 'true'

# Driver program

if \_\_name\_\_ == "\_\_main\_\_":

    input ='GEEGEEKSKS'

    pattern ='GEEKS'

    print checkEmpty(input, pattern)

# String slicing in Python to rotate a string

Input : s = "GeeksforGeeks"

d = 2

Output : Left Rotation : "eksforGeeksGe"

Right Rotation : "ksGeeksforGee"

# Function to rotate string left and right by d length

def rotate(input,d):

    # slice string in two parts for left and right

    Lfirst = input[0 : d]

    Lsecond = input[d :]

    Rfirst = input[0 : len(input)-d]

    Rsecond = input[len(input)-d : ]

    # now concatenate two parts together

    print "Left Rotation : ", (Lsecond + Lfirst)

    print "Right Rotation : ", (Rsecond + Rfirst)

# Driver program

if \_\_name\_\_ == "\_\_main\_\_":

    input = 'GeeksforGeeks'

    d=2

    rotate(input,d)

# Python | Check for URL in a String

|  |
| --- |
| # Python code to find the URL from an input string  # Using the regular expression  import re    def Find(string):      # findall() has been used      # with valid conditions for urls in string      url = re.findall('http[s]?://(?:[a-zA-Z]|[0-9]|[$-\_@.&+]      |[!\*\(\), ]|(?:%[0-9a-fA-F][0-9a-fA-F]))+', string)      return url    # Driver Code  string = 'My Profile: https://auth.geeksforgeeks.org  / user / Chinmoy % 20Lenka / articles in  the portal of http://www.geeksforgeeks.org/'  print("Urls: ", Find(string)) |

Output:

Urls: ['https://auth.geeksforgeeks.org/user/Chinmoy%20Lenka/articles',

'http://www.geeksforgeeks.org/']

# Python | Check if a given string is binary string or not

Input: str = "01010101010"

Output: Yes

|  |
| --- |
| Python program to check  # if a string is binary or not    # function for checking the  # string is accepted or not  def check(string) :        # set function convert string      # into set of characters .      p = set(string)        # declare set of '0', '1' .      s = {'0', '1'}        # check set p is same as set s      # or set p contains only '0'      # or set p contains only '1'      # or not, if any one conditon      # is true then string is accepted      # otherwise not .      if s == p or p == {'0'} or p == {'1'}:          print("Yes")      else :          print("No")        # driver code  if \_\_name\_\_ == "\_\_main\_\_" :        string = "101010000111"        # function calling      check(string) |

**Output:**

Yes