1. **Python – Replace multiple words with K**

# Python3 code to demonstrate working of

# Replace multiple words with K

# Using regex + join()

import re

# initializing string

test\_str = 'Geeksforgeeks is best for geeks and CS'

# printing original string

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

# initializing word list

word\_list = ["best", 'CS', 'for']

# initializing replace word

repl\_wrd = 'gfg'

# Replace multiple words with K

# Using regex + join()

res = re.sub("|".join(sorted(word\_list, key = len, reverse = True)), repl\_wrd, test\_str)

# printing result

print("String after multiple replace : " + res)

**output:**

The original string is: Geeksforgeeks is best for geeks and CS

String after multiple replace: Geeksgfggeeks is gfg gfg geeks and gfg

1. **Python | Permutation of a given string using inbuilt function**

# Function to find permutations of a given string

from itertools import permutations

def allPermutations(str):

  # Get all permutations of string 'ABC'

  permList = permutations(str)

  # print all permutations

  for perm in list(permList):

    print (''.join(perm))

# Driver program

if \_\_name\_\_ == "\_\_main\_\_":

  str = 'ABC'

  allPermutations(str)

**Output:** ABC

ACB

BAC

BCA

CAB

CBA

1. **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

  regex = r"(?i)\b((?:https?://|www\d{0,3}[.]|[a-z0-9.\-]+[.][a-z]{2,4}/)(?:[^\s()<>]+|\(([^\s()<>]+|(\([^\s()<>]+\)))\*\))+(?:\(([^\s()<>]+|(\([^\s()<>]+\)))\*\)|[^\s`!()\[\]{};:'\".,<>?«»“”‘’]))"

  url = re.findall(regex, string)

  return [x[0] for x in url]

# Driver Code

string = 'My Profile: https://auth.geeksforgeeks.org/user/Chinmoy%20Lenka/articles in the portal of https://www.geeksforgeeks.org/'

print("Urls: ", Find(string))

**Output:** Urls: ['<https://auth.geeksforgeeks.org/user/Chinmoy%20Lenka/articles>', '<https://www.geeksforgeeks.org/>']

1. **Execute a String of Code in Python**

# Python program to illustrate use of exec to

# execute a given code as string.

# function illustrating how exec() functions.

def exec\_code():

  LOC = """

def factorial(num):

  fact=1

  for i in range(1,num+1):

    fact = fact\*i

  return fact

print(factorial(5))

"""

  exec(LOC)

# Driver Code

exec\_code()

**Output:** 120

1. **String slicing in Python to rotate a string**

# 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)

**Output:** Left Rotation: eksforGeeksGe

Right Rotation: ksGeeksforGee

1. **String slicing in Python to check if a string can become empty by recursive deletion**

def checkEmpty(input, pattern):

  # If both are empty

  if len(input)== 0 and len(pattern)== 0:

    return 'true'

  # If only pattern is empty

  if len(pattern)== 0:

    return 'true'

  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))

**Output:** true

1. **Python Counter| Find all duplicate characters in string**

from collections import Counter

def find\_dup\_char(input):

  # now create dictionary using counter method

  # which will have strings as key and their

  # frequencies as value

  WC = Counter(input)

  # Finding no. of occurrence of a character

  # and get the index of it.

  for letter, count in WC.items():

    if (count > 1):

      print(letter)

# Driver program

if \_\_name\_\_ == "\_\_main\_\_":

  input = 'geeksforgeeks'

  find\_dup\_char(input)

**Output:**

g

e

k

s

1. **Python – Replace all occurrences of a substring in a string**

#Python has inbuilt function replace to replace all occurrences of substring.

input\_string = "geeksforgeeks"

s1 = "geeks"

s2 = "abcd"

input\_string = input\_string.replace(s1, s2)

print(input\_string)

**Output:** abcdforabcd

1. **Python – Extract Unique values dictionary values**
2. **Python program to find the sum of all items in a dictionary**