1. **Python – Least Frequent Character in String**

# Python 3 code to demonstrate

# Least Frequent Character in String

# collections.Counter() + min()

from collections import Counter

# initializing string

test\_str = "GeeksforGeeks"

# printing original string

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

# using collections.Counter() + min() to get

# Least Frequent Character in String

res = Counter(test\_str)

res = min(res, key = res.get)

# printing result

print ("The minimum of all characters in GeeksforGeeks is : " + res)

**Output**: The original string is : GeeksforGeeks

The minimum of all characters in GeeksforGeeks is : f

1. **Python | Maximum frequency character in String**

from collections import Counter

# initializing string

test\_str = "GeeksforGeeks"

# printing original string

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

# using collections.Counter() + max() to get

# Maximum frequency character in String

res = Counter(test\_str)

res = max(res, key = res.get)

# printing result

print ("The maximum of all characters in GeeksforGeeks is : " + str(res))

**Output:** The original string is : GeeksforGeeks

The maximum of all characters in GeeksforGeeks is : e

1. **Python | Program to check if a string contains any special character**

# Python3 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)

**Output:** String is not accepted.

1. **Generating random strings until a given string is generated**

# Python program to generate and match

# the string from all random strings

# of same length

# Importing string, random

# and time modules

import string

import random

import time

# all possible characters including

# lowercase, uppercase and special symbols

possibleCharacters = string.ascii\_lowercase + string.digits + string.ascii\_uppercase + ' ., !?;:'

# string to be generated

t = "geek"

# To take input from user

# t = input(str("Enter your target text: "))

attemptThis = ''.join(random.choice(possibleCharacters)

                for i in range(len(t)))

attemptNext = ''

completed = False

iteration = 0

# Iterate while completed is false

while completed == False:

  print(attemptThis)

  attemptNext = ''

  completed = True

  # Fix the index if matches with

  # the strings to be generated

  for i in range(len(t)):

    if attemptThis[i] != t[i]:

      completed = False

      attemptNext += random.choice(possibleCharacters)

    else:

      attemptNext += t[i]

  # increment the iteration

  iteration += 1

  attemptThis = attemptNext

  time.sleep(0.1)

# Driver Code

print("Target matched after " +

  str(iteration) + " iterations")

1. **Find words which are greater than given length k**

# Python program to find all string

# which are greater than given length k

# function find string greater than length k

def string\_k(k, str):

  # create the empty string

  string = []

  # split the string where space is comes

  text = str.split(" ")

  # iterate the loop till every substring

  for x in text:

    # if length of current sub string

    # is greater than k then

    if len(x) > k:

      # append this sub string in

      # string list

      string.append(x)

  # return string list

  return string

# Driver Program

k = 3

str ="geek for geeks"

print(string\_k(k, str))

**Output:** ['geek', 'geeks']

1. **Python program for removing i-th character from a string**

# Python3 program for removing i-th

# indexed character from a string

# Removes character at index i

def remove(string, i):

  # Characters before the i-th indexed

  # is stored in a variable a

  a = string[: i]

  # Characters after the nth indexed

  # is stored in a variable b

  b = string[i + 1:]

  # Returning string after removing

  # nth indexed character.

  return a + b

# Driver Code

if \_\_name\_\_ == '\_\_main\_\_':

  string = "geeksFORgeeks"

  # Remove nth index element

  i = 5

  # Print the new string

  print(remove(string, i))

**Output:** geeksORgeeks

1. **Python program to split and join a string**

# Python program to split a string and

# join it using different delimiter

def split\_string(string):

  # Split the string based on space delimiter

  list\_string = string.split(' ')

  return list\_string

def join\_string(list\_string):

  # Join the string based on '-' delimiter

  string = '-'.join(list\_string)

  return string

# Driver Function

if \_\_name\_\_ == '\_\_main\_\_':

  string = 'Geeks for Geeks'

  # Splitting a string

  list\_string = split\_string(string)

  print(list\_string)

  # Join list of strings into one

  new\_string = join\_string(list\_string)

  print(new\_string)

**Output:** ['Geeks', 'for', 'Geeks']

Geeks-for-Geeks

1. **Python | Check if a given string is binary string 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 condition

  # 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

1. **Python program to find uncommon words from two Strings**

def UncommonWords(A, B):

  # count will contain all the word counts

  count = {}

  # insert words of string A to hash

  for word in A.split():

    count[word] = count.get(word, 0) + 1

  # insert words of string B to hash

  for word in B.split():

    count[word] = count.get(word, 0) + 1

  # return required list of words

  return [word for word in count if count[word] == 1]

# Driver Code

A = "Geeks for Geeks"

B = "Learning from Geeks for Geeks"

# Print required answer

print(UncommonWords(A, B))

**Output:** ['Learning', 'from']

1. **Python – Replace duplicate Occurrence in String**

# Python3 code to demonstrate working of

# Replace duplicate Occurrence in String

# Using split() + enumerate() + loop

# initializing string

test\_str = 'Gfg is best . Gfg also has Classes now. \

        Classes help understand better . '

# printing original string

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

# initializing replace mapping

repl\_dict = {'Gfg' : 'It', 'Classes' : 'They' }

# Replace duplicate Occurrence in String

# Using split() + enumerate() + loop

test\_list = test\_str.split(' ')

res = set()

for idx, ele in enumerate(test\_list):

  if ele in repl\_dict:

    if ele in res:

      test\_list[idx] = repl\_dict[ele]

    else:

      res.add(ele)

res = ' '.join(test\_list)

# printing result

print("The string after replacing : " + res)

**Output:**

The original string is : Gfg is best . Gfg also has Classes now. Classes help understand better .

The string after replacing : Gfg is best . It also has Classes now. Classes help understand better **.**