1. Write a Python program to calculate the length of a string.

ANS:

string = input("Enter a string: ") length = len(string)

print("Length of the string:", length)

1. Write a Python program to count the number of characters (character frequency) in a string. Sample String : [google.com](http://google.com/)'

Expected Result : {'o': 3, 'g': 2, '.': 1, 'e': 1, 'l': 1, 'm': 1, 'c': 1}

ANS:

string = 'google.com' char\_frequency = {} for char in string:

if char in char\_frequency: char\_frequency[char] += 1

else:

char\_frequency[char] = 1 print("Character frequency:", char\_frequency)

1. Write a Python program to get a string made of the ﬁrst 2 and the last 2 chars from a given a string. If the string length is less than 2, return instead of the empty string.

Sample String : 'thisisniceone' Expected Result : 'thne”' Sample String : 'ab'

Expected Result : 'abab' Sample String : 'f'

Expected Result : Empty String ANS:

1. Write a Python program to get a string from a given string where all occurrences of its ﬁrst char have been changed to '$', except the ﬁrst char itself.

Sample String : 'restart' Expected Result : 'resta$t'

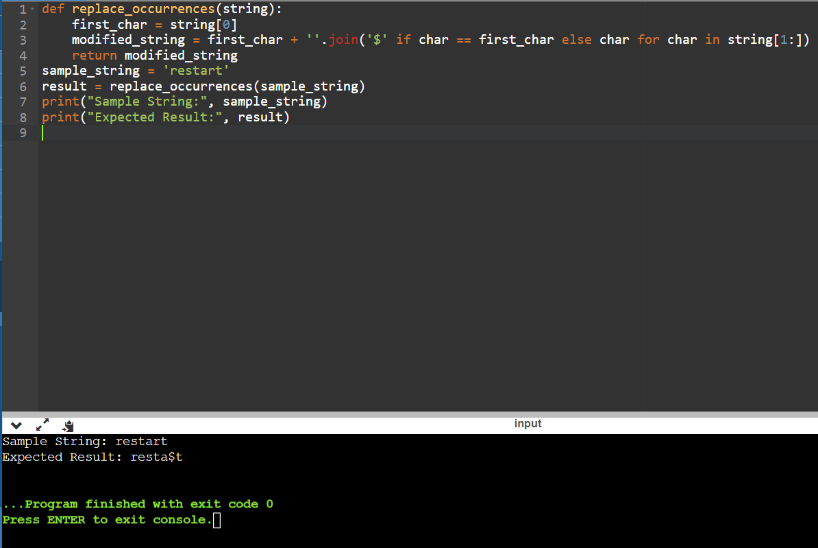
ANS:

def replace\_occurrences(string): first\_char = string[0]

modified\_string = first\_char + ''.join('$' if char == first\_char else char for char in string[1:])

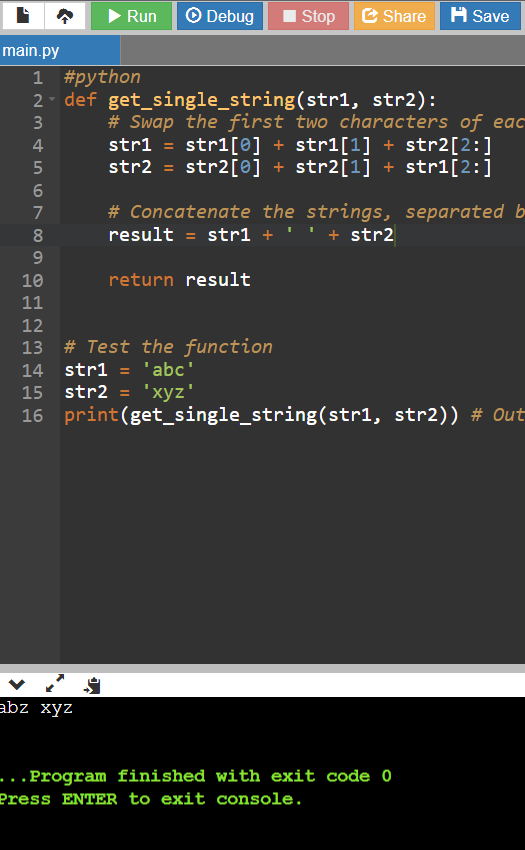
return modified\_string sample\_string = 'restart'

result = replace\_occurrences(sample\_string) print("Sample String:", sample\_string) print("Expected Result:", result)



1. Write a Python program to get a single string from two given strings, separated by a space and swap the ﬁrst two characters of each string.

Sample String : 'abc', 'xyz' Expected Result : 'xyc abz'



1. Write a Python program to add 'ing' at the end of a given string (length should be at least 3). If the given string already ends with 'ing' then add 'ly' instead. If the string length of the given string is less than 3, leave it unchanged.

Sample String : 'abc' Expected Result : 'abcing' Sample String : 'string' Expected Result : 'stringly'

ANS:

def modify\_string(string): if len(string) < 3:

return string

elif string[-3:] == 'ing': return string + 'ly'

else:

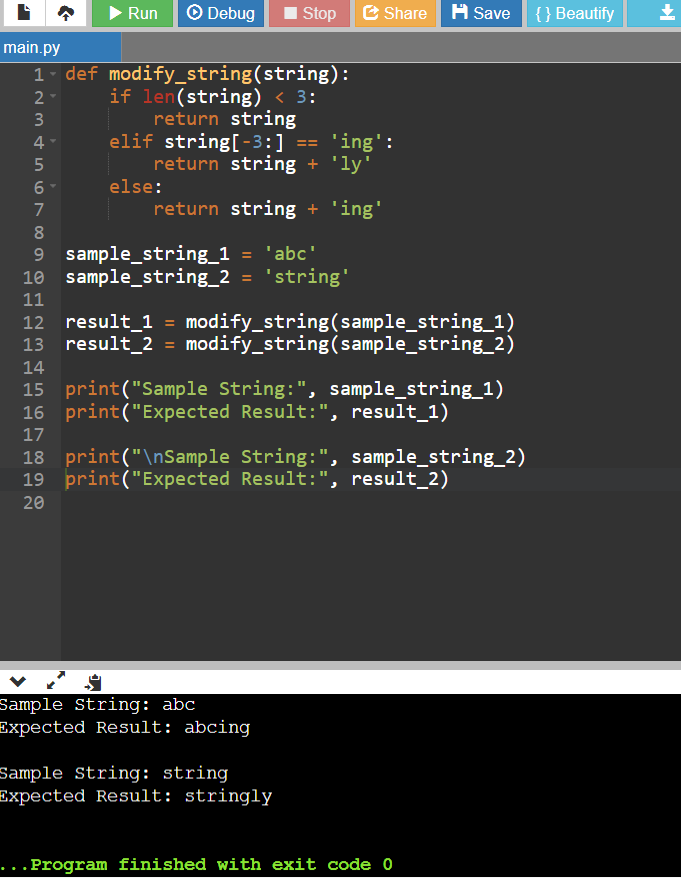
return string + 'ing'

sample\_string\_1 = 'abc' sample\_string\_2 = 'string'

result\_1 = modify\_string(sample\_string\_1) result\_2 = modify\_string(sample\_string\_2)

print("Sample String:", sample\_string\_1) print("Expected Result:", result\_1)

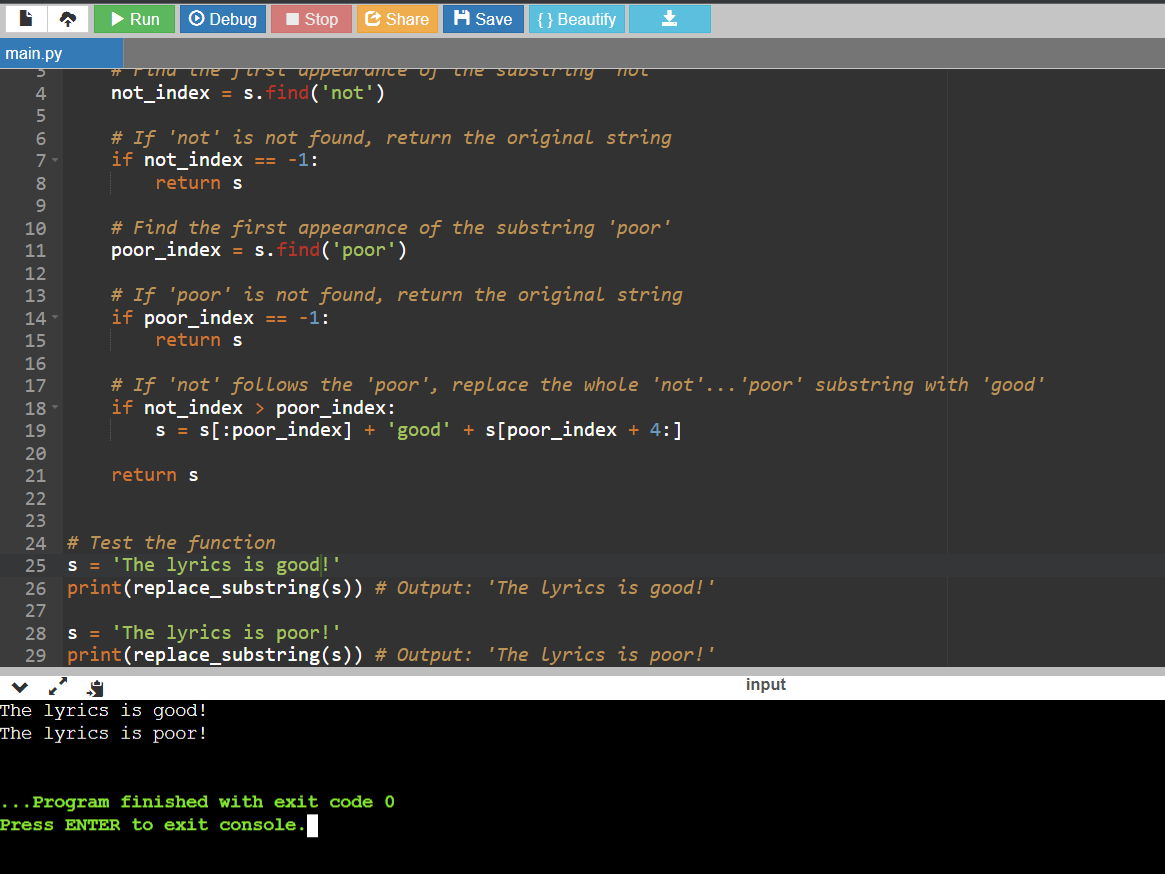
print("\nSample String:", sample\_string\_2) print("Expected Result:", result\_2)



1. Write a Python program to ﬁnd the ﬁrst appearance of the substring 'not' and 'poor' from a given string, if 'not' follows the 'poor', replace the whole 'not'...'poor' substring with 'good'. Return the resulting string.

Sample String : 'The lyrics is not that poor!' 'The lyrics is poor!'

Expected Result : 'The lyrics is good!' 'The lyrics is poor!'



1. Write a Python function that takes a list of words and returns the length of the longest one.

ANS:

def find\_longest\_word():

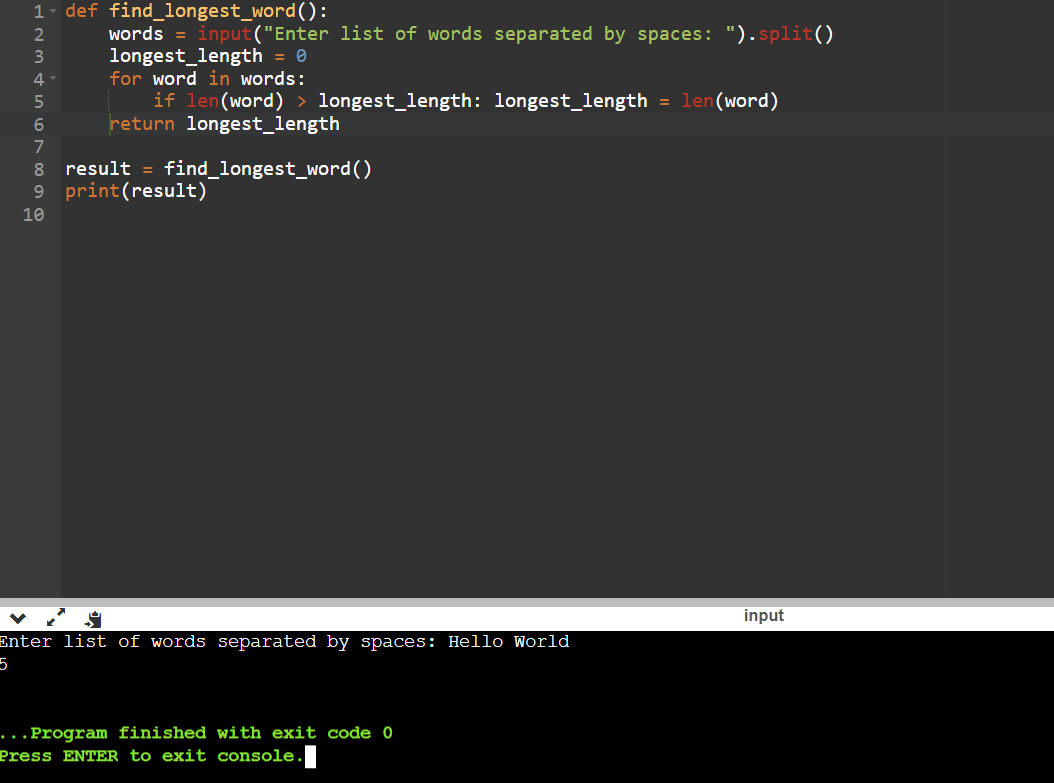
words = input("Enter list of words separated by spaces: ").split() longest\_length = 0

for word in words:

if len(word) > longest\_length: longest\_length = len(word)

return longest\_length

result = find\_longest\_word() print(result)



1. Write a Python program to remove the nth index character from a nonempty string.

ANS:

def remove\_nth\_character(string, n): if n < 0 or n >= len(string):

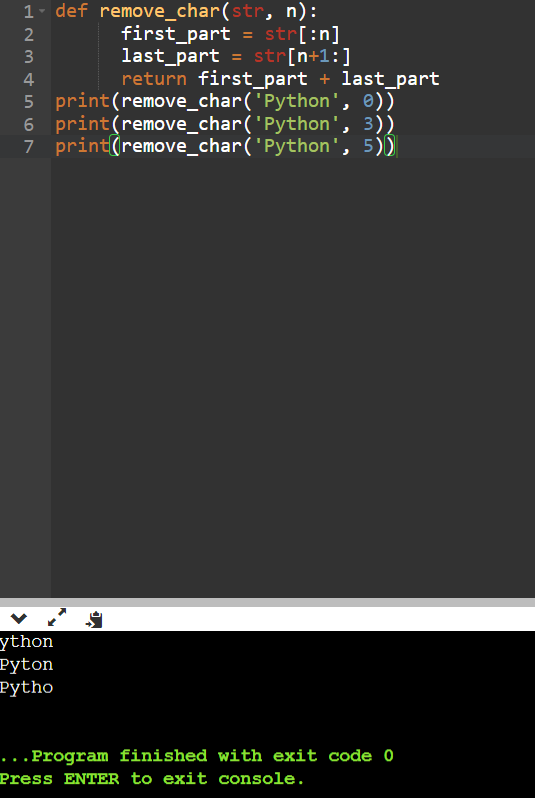
return "index not valid" return string[:n] + string[n+1:]

word = input("Enter a word: ")

index = int(input("Enter the index of the character to be removed: ")) new\_word = remove\_nth\_character(word, index)

print("Original Word:", word)

print("Word after removing the character at index", index, ":", new\_word)



1. Write a Python program that accepts a comma separated sequence of words as input and prints the unique words in sorted form (alphanumerically).

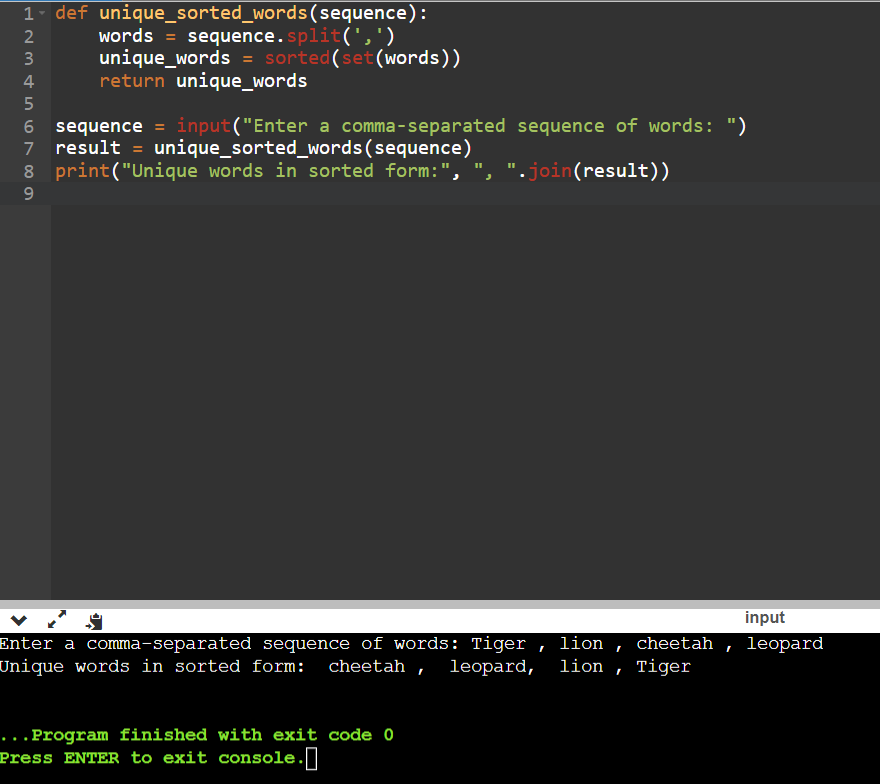
Sample Words : red, white, black, red, green, black Expected Result : black, green, red, white

ANS:

def unique\_sorted\_words(sequence): words = sequence.split(',') unique\_words = sorted(set(words)) return unique\_words

sequence = input("Enter a comma-separated sequence of words: ") result = unique\_sorted\_words(sequence)

print("Unique words in sorted form:", ", ".join(result))



1. Write a Python function to reverses a string if it's length is a multiple of 4.

ANS:

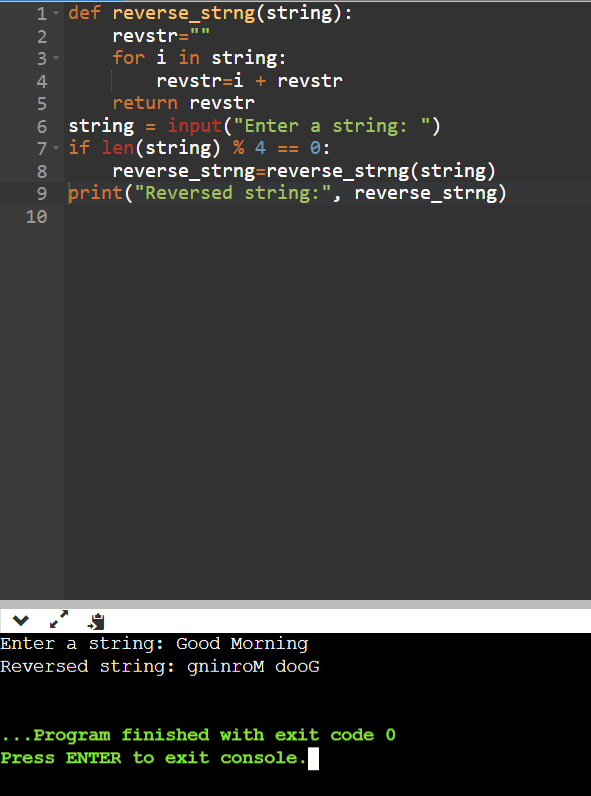
def reverse\_strng(string): revstr=""

for i in string: revstr=i + revstr

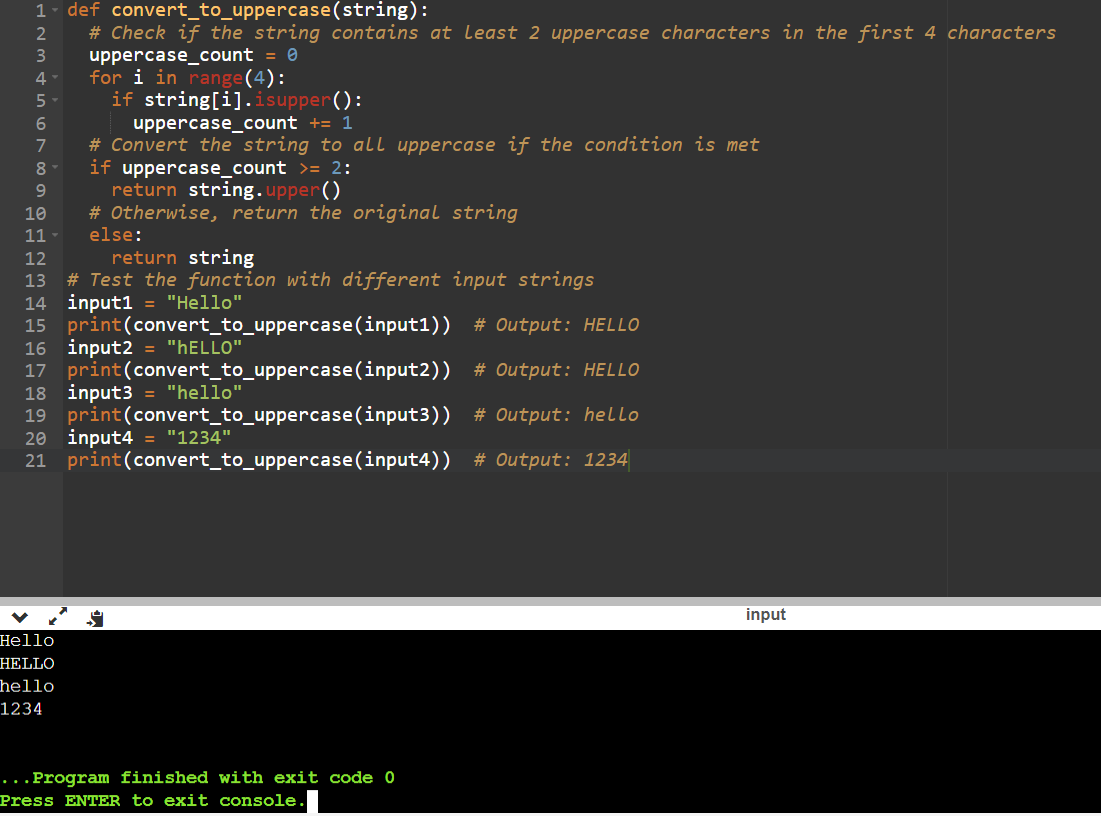
return revstr

string = input("Enter a string: ") if len(string) % 4 == 0:

reverse\_strng=reverse\_strng(string) print("Reversed string:", reverse\_strng)



1. Write a Python function to convert a given string to all uppercase if it contains at least 2 uppercase characters in the ﬁrst 4 characters.



1. Write a Python program to check whether a string starts with speciﬁed characters.

ANS:

def starts\_with(string, preﬁx): return string[:len(preﬁx)] == preﬁx

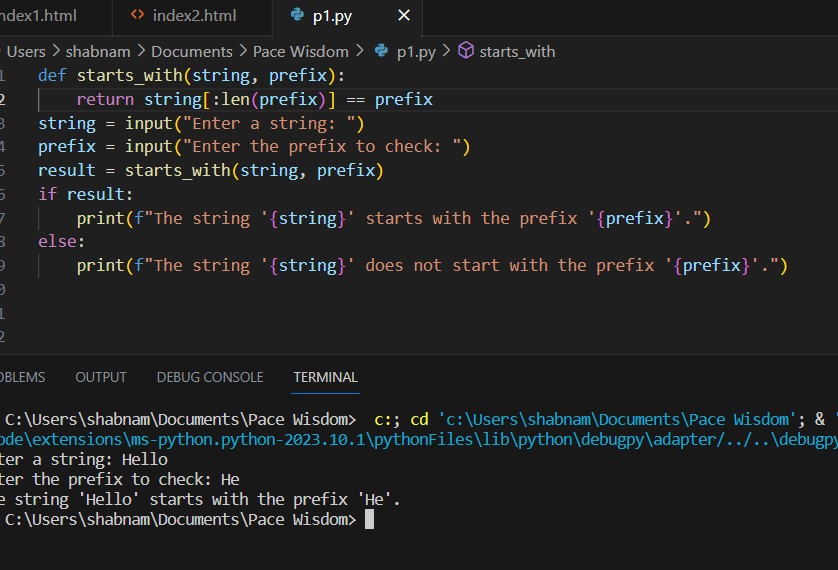
string = input("Enter a string: ")

preﬁx = input("Enter the preﬁx to check: ") result = starts\_with(string, preﬁx)

if result:

print(f"The string '{string}' starts with the preﬁx '{preﬁx}'.") else:

print(f"The string '{string}' does not start with the preﬁx '{preﬁx}'.")

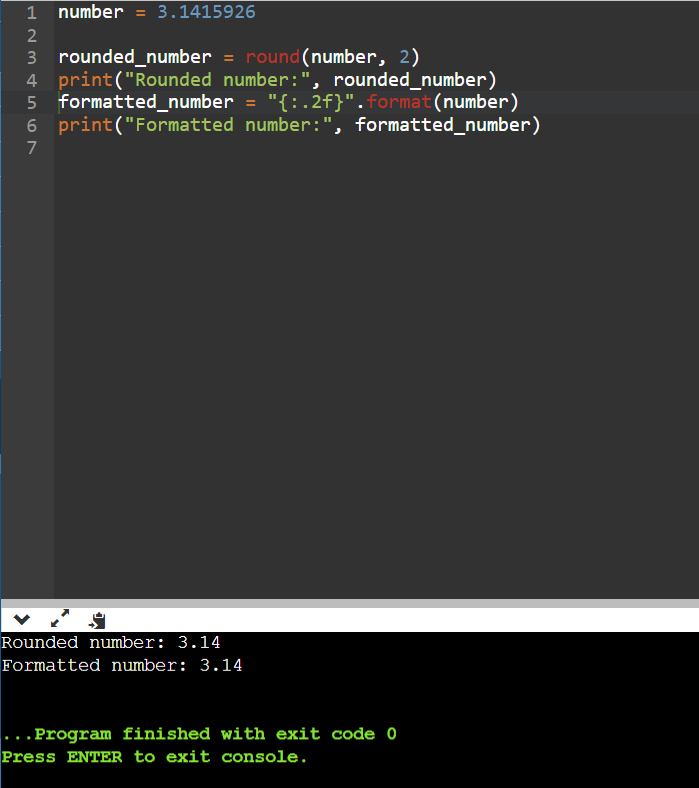


1. Write a Python program to print the following ﬂoating numbers upto 2 decimal places. 3.1415926

ANS:

number = 3.1415926

rounded\_number = round(number, 2) print("Rounded number:", rounded\_number) formatted\_number = "{:.2f}".format(number) print("Formatted number:", formatted\_number)



1. Write a Python program to count repeated characters in a string. Sample string: 'thequickbrownfoxjumpsoverthelazydog'

Expected output :

o 4

e 3

u 2

h 2

r 2

t 2

ANS:

def repeated\_character(string): char\_count = {}

for char in string:

if char in char\_count: char\_count[char] += 1

else:

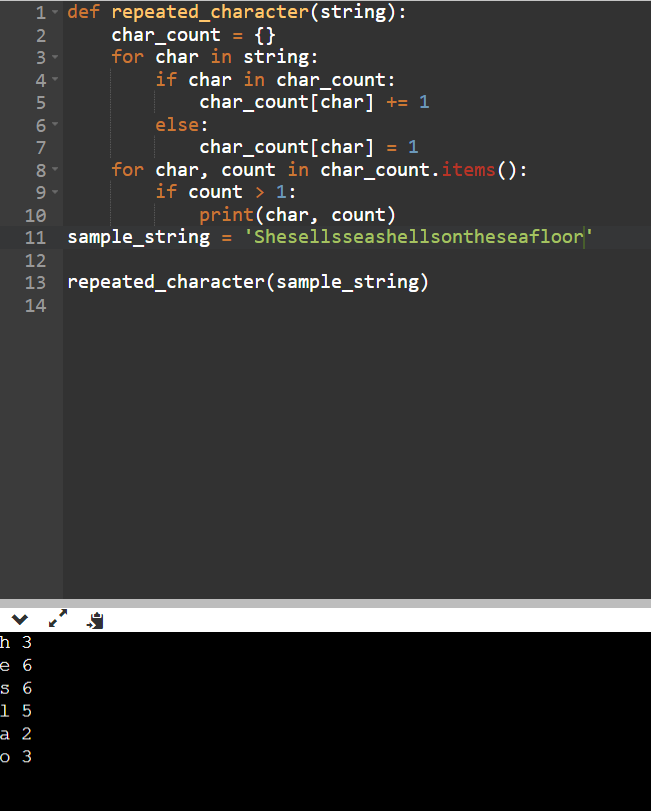
char\_count[char] = 1

for char, count in char\_count.items(): if count > 1:

print(char, count)

sample\_string = 'thequickbrownfoxjumpsoverthelazydog'

repeated\_character(sample\_string)

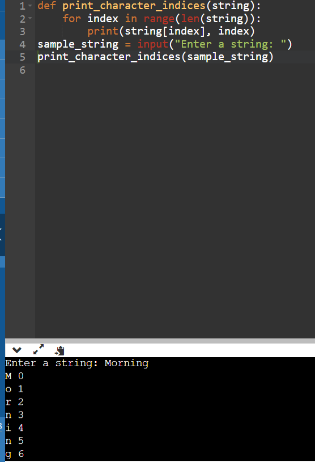


1. Write a Python program to print the index of the character in a string.

ANS:

def print\_character\_indices(string): for index in range(len(string)): print(string[index], index)

sample\_string = input("Enter a string: ") print\_character\_indices(sample\_string)



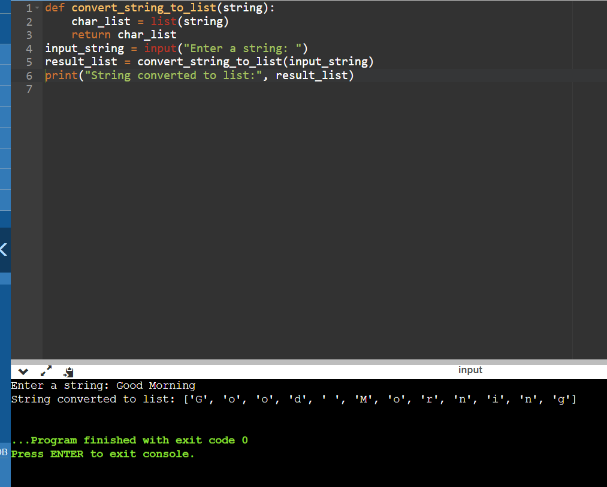
1. Write a Python program to convert a string in a list.

ANS:

def convert\_string\_to\_list(string): char\_list = list(string)

return char\_list

input\_string = input("Enter a string: ") result\_list = convert\_string\_to\_list(input\_string) print("String converted to list:", result\_list)



1. Write a Python program to swap comma and dot in a string. Sample string: "32.054,23"

Expected Output: "32,054.23" ANS:

def swap(string): swapped\_string = "" for char in string:

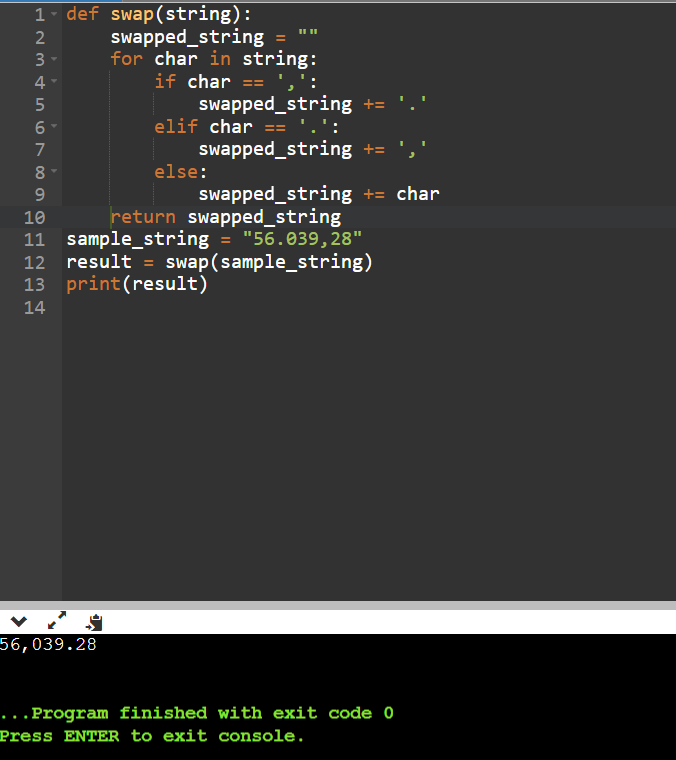
if char == ',': swapped\_string += '.'

elif char == '.': swapped\_string += ','

else:

swapped\_string += char return swapped\_string

sample\_string = "32.054,23" result = swap(sample\_string) print(result)



1. Write a Python program to ﬁnd smallest and largest word in a given string.

ANS:

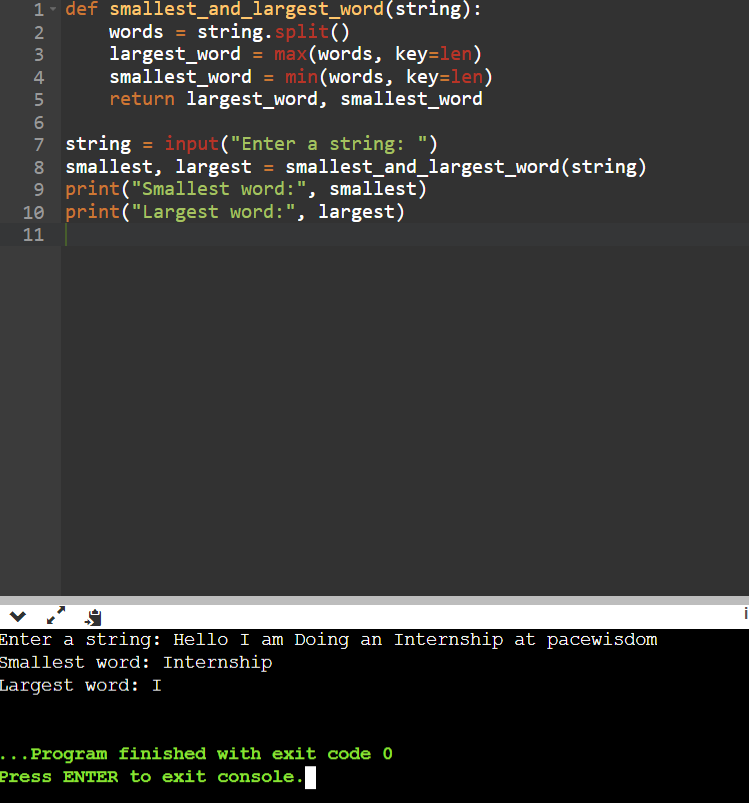
def smallest\_and\_largest\_word(string): words = string.split()

largest\_word = max(words, key=len) smallest\_word = min(words, key=len) return largest\_word, smallest\_word

string = input("Enter a string: ")

smallest, largest = smallest\_and\_largest\_word(string) print("Smallest word:", smallest)

print("Largest word:", largest)



1. Write a Python program to remove all consecutive duplicates of a given string.

ANS:

def remove\_consecutive\_duplicates(string): result = ""

previous\_char = None for char in string:

if char != previous\_char: result += char

previous\_char = char return result

sample\_string = input("Enter a string: ") print(remove\_consecutive\_duplicates(sample\_string))

