**6/6**

1. Создайте функцию, которая принимает две строки. Первая строка содержит предложение, содержащее буквы второй строки в последовательной последовательности, но в другом порядке. Скрытая анаграмма должна содержать все буквы, включая дубликаты, из второй строки в любом порядке и не должна содержать никаких других букв алфавита.

Напишите функцию, чтобы найти анаграмму второй строки, вложенную где-то в первую строку. Вы должны игнорировать регистр символов, любые пробелы и знаки препинания и возвращать анаграмму в виде строчной строки без пробелов или знаков препинания.

Пример:

hiddenAnagram(["My world evolves in a beautiful space called Tesh.", "sworn love lived"]) ➞ "worldevolvesin"

// The sequence "world evolves in" is a perfect anagram of "sworn love lived".

hiddenAnagram("An old west action hero actor", "Clint Eastwood") ➞ "noldwestactio"

// The sequence "n old west actio" is a perfect anagram of "Clint Eastwood".

hiddenAnagram("Mr. Mojo Rising could be a song title", "Jim Morrison") ➞ "mrmojorisin"

// The sequence "Mr. Mojo Risin" ignoring the full stop, is a perfect

// anagram of "Jim Morrison".

hiddenAnagram("Banana? margaritas", "ANAGRAM") ➞ "anamarg"

// The sequence "ana? marg" ignoring "?" is a perfect anagram of "Anagram".

hiddenAnagram("D e b90it->?$ (c)a r...d,,#~", "bad credit") ➞ "debitcard"

// When all spaces, numbers and puntuation marks are removed

// from the whole phrase, the remaining characters form the sequence

// "Debitcard" which is a perfect anagram of "bad credit".

hiddenAnagram("Bright is the moon", "Bongo mirth") ➞ "notfound"

// The words "Bright moon" are an anagram of "bongo mirth" but they are

// not a continuous alphabetical sequence because the words "is the" are in

// between. Hence the negative result, "notfound" is returned.

Примечание:

- Совершенная анаграмма содержит все буквы оригинала в любом порядке, ни больше, ни меньше.

- Если в предложении нет скрытой анаграммы, верните "notfound".

- Как и в приведенных выше примерах, скрытая анаграмма может начинаться или заканчиваться частично через слово и/или охватывать несколько слов и может содержать знаки препинания и другие не-альфа-символы.

- Игнорируйте регистр символов и любые встроенные не-альфа-символы.

- Если в предложении больше 1 результата, верните ближайший к началу.

1. Напишите функцию, которая возвращает массив строк, заполненных из срезов символов n-длины данного слова (срез за другим, в то время как n-длина применяется к слову).

Пример:

collect("intercontinentalisationalism", 6)

➞ ["ationa", "interc", "ntalis", "ontine"]

collect("strengths", 3)

➞ ["eng", "str", "ths"]

collect("pneumonoultramicroscopicsilicovolcanoconiosis", 15)

➞ ["croscopicsilico", "pneumonoultrami", "volcanoconiosis"]

Примечания:

- Убедитесь, что результирующий массив лексикографически упорядочен.

- Возвращает пустой массив, если заданная строка меньше n.

- Ожидается, что вы решите эту задачу с помощью рекурсивного подхода.

1. В шифре Nico кодирование осуществляется путем создания цифрового ключа и присвоения каждой буквенной позиции сообщения с помощью предоставленного ключа.

Создайте функцию, которая принимает два аргумента, message и key, и возвращает закодированное сообщение.

Существуют некоторые вариации правил шифрования. Одна из версий правил шифрования изложена ниже:

message = "mubashirhassan"

key = "crazy"

nicoCipher(message, key) ➞ "bmusarhiahass n"

Шаг 1: Назначьте числа отсортированным буквам из ключа:

"crazy" = 23154

Шаг 2: Назначьте номера буквам данного сообщения:

2 3 1 5 4

---------

m u b a s

h i r h a

s s a n

Шаг 3: Сортировка столбцов по назначенным номерам:

1 2 3 4 5

---------

b m u s a

r h i a h

a s s n

Шаг 4: Верните закодированное сообщение по строкам:

eMessage = "bmusarhiahass n"

Пример:

nicoCipher("myworldevolvesinhers", "tesh") ➞ "yowmledrovlvsnieesrh"

nicoCipher("andiloveherso", "tesha") ➞ "lnidaevheo s or"

nicoCipher("mubashirhassan", "crazy") ➞ "bmusarhiahass n"

nicoCipher("edabitisamazing", "matt") ➞ "deabtiismaaznig "

nicoCipher("iloveher", "612345") ➞ "lovehir e"

1. Создайте метод, который принимает массив arr и число n и возвращает массив из двух целых чисел из arr, произведение которых равно числу n следующего вида:

[value\_at\_lower\_index, value\_at\_higher\_index]

Убедитесь, что вы возвращаете массив из двух целых чисел, который точно делит n (произведение n) и что индексы совпадают с указанным выше форматом. Таким образом, сортировка значений основана на восходящих индексах.

Пример:

twoProduct([1, 2, 3, 9, 4, 5, 15], 45) ➞ [9, 5]

// at index 5 which has the value 5 is a full match

// to the value at index 3 which is 9

// the closest gap between indices that equates

// to the product which is 45

twoProduct([1, 2, 3, 9, 4, 15, 3, 5], 45) ➞ [3, 15]

// at index 5 which has the value 15 is a full match

// to the value at index 2 which is 3

// the closest gap between indices that equates

// to the product which is 45

twoProduct([1, 2, -1, 4, 5, 6, 10, 7], 20) ➞ [4, 5]

// at index 4 which has the value 5 is a full match

// to the value at index 3 which is 4

// a full match can only be achieved if you've found the multiplier at an

// index lower than the current index, thus, 5 (@ index 4) has a pair lower

// than its index which is the value 4 (@ index 3), so, 5\*4 = 20, in which 5

// has a higher index (4) than 4 which has an index value of 3

twoProduct([1, 2, 3, 4, 5, 6, 7, 8, 9, 10], 10) ➞ [2, 5]

twoProduct([100, 12, 4, 1, 2], 15) ➞ []

Примечание:

- Дубликатов не будет.

- Возвращает пустой массив, если совпадение не найдено.

- Всегда считайте, что пара рассматриваемого элемента (текущий элемент, на который указывали во время поиска) находится слева от него.

- Массив может иметь несколько решений (произведений n), поэтому возвращайте первое найденное полное совпадение (как описано выше).

1. Создайте рекурсивную функцию, которая проверяет, является ли число точной верхней границей факториала n. Если это так, верните массив точной факториальной границы и n, или иначе, пустой массив.

Пример:

isExact(6) ➞ [6, 3]

isExact(24) ➞ [24, 4]

isExact(125) ➞ []

isExact(720) ➞ [720, 6]

isExact(1024) ➞ []

isExact(40320) ➞ [40320, 8]

1. Деление на дробь часто приводит к бесконечно повторяющейся десятичной дроби.

1/3=.3333333... 1/7=.142857142857...

Создайте функцию, которая принимает десятичную дробь в строковой форме с повторяющейся частью в круглых скобках и возвращает эквивалентную дробь в строковой форме и в наименьших членах.

Пример:

fractions("0.(6)") ➞ "2/3"

fractions("1.(1)") ➞ "10/9"

fractions("3.(142857)") ➞ "22/7"

fractions("0.19(2367)") ➞ "5343/27775"

fractions("0.1097(3)") ➞ "823/7500"

1. В этой задаче преобразуйте строку в серию слов (или последовательности символов), разделенных одним пробелом, причем каждое слово имеет одинаковую длину, заданную первыми 15 цифрами десятичного представления числа Пи:

3.14159265358979

Если строка содержит больше символов, чем общее количество, заданное суммой цифр Пи, неиспользуемые символы отбрасываются, и вы будете использовать только те, которые необходимы для формирования 15 слов.

String = "HOWINEEDADRINKALCOHOLICINNATUREAFTERTHEHEAVYLECTURESINVOLVINGQUANTUMMECHANICSANDALLTHESECRETSOFTHEUNIVERSE"

Pi String = "HOW I NEED A DRINK ALCOHOLIC IN NATURE AFTER THE HEAVY LECTURES INVOLVING QUANTUM MECHANICS"

// Every word has the same length of the digit of Pi found at the same index ?

// "HOW" = 3, "I" = 1, "NEED" = 4, "A" = 1, "DRINK" = 5

// "ALCOHOLIC" = 9, "IN" = 2, "NATURE" = 6, "AFTER" = 5

// "THE" = 3, "HEAVY" = 5, "LECTURES" = 8, "INVOLVING" = 9

// "QUANTUM" = 7, "MECHANICS" = 9

// 3.14159265358979

Также, если строка содержит меньше символов, чем общее количество, заданное суммой цифр Пи, в любом случае вы должны соблюдать последовательность цифр для получения слов.

String = "FORALOOP"

Pi String = "FOR A LOOP"

// Every word has the same length of the digit of Pi found at the same index ?

// "FOR" = 3, "A" = 1, "LOOP" = 4

// 3.14

Если буквы, содержащиеся в строке, не совпадают в точности с цифрами, в этом случае вы будете повторять последнюю букву, пока слово не будет иметь правильную длину.

String = "CANIMAKEAGUESSNOW"

Pi String = "CAN I MAKE A GUESS NOWWWWWWW"

// Every word has the same length of the digit of Pi found at the same index ?

// "CAN" = 3, "I" = 1, "MAKE" = 4, "A" = 1, "GUESS" = 5, "NOW" = 3

// 3.14153 (Wrong!)

// The length of the sixth word "NOW" (3)...

// ...doesn't match the sixth digit of Pi (9)

// The last letter "W" will be repeated...

// ...until the length of the word will match the digit

// "CAN" = 3, "I" = 1, "MAKE" = 4, "A" = 1, "GUESS" = 5, "NOWWWWWWW" = 9

// 3.14159 (Correct!)

Если данная строка пуста, должна быть возвращена пустая строка.

Учитывая строку txt, реализуйте функцию, которая возвращает ту же строку, отформатированную в соответствии с приведенными выше инструкциями.

Пример:

pilish\_string("33314444") ➞ "333 1 4444"

// 3.14

pilish\_string("TOP") ➞ "TOP"

// 3

pilish\_string("X")➞ "XXX"

// "X" has to match the same length of the first digit (3)

// The last letter of the word is repeated

pilish\_string("")➞ ""

Примечание:

- Эта задача представляет собой упрощенную концепцию, вдохновленную Пилишем, своеобразным типом ограниченного письма, в котором используются все известные возможные цифры Пи. Потенциально бесконечный текст может быть написан с использованием знаков препинания и набора дополнительных правил, которые позволяют избежать длинных последовательностей маленьких цифр, заменяя их словами больше 9 букв и делая так, чтобы композиция выглядела более похожей на стихотворение вольным стихом.

- Точка, отделяющая целую часть числа Пи от десятичной, не должна учитываться в функции: она присутствует в инструкциях и примерах только для удобства чтения.

1. Создайте функцию, которая будет вычислять результат математических выражений, предоставленных в виде строки.

Реализуйте алгоритм, который разбирает строку и вычисляет результат выражения, учитывая приоритет операций, скобки и т. д. Математические операции, которые нужно поддерживать, включают в себя сложение, вычитание, умножение, деление и скобки. Обработайте ошибки, такие как деление на ноль или неправильно введенное выражение, и верните соответствующее сообщение об ошибке.

Пример:

generateNonconsecutive("3 + 5 \* (2 - 6)") ➞ -17

generateNonconsecutive("6 – 18 / (-1 + 4)") ➞ 0

1. Шерлок считает строку действительной, если все символы строки встречаются одинаковое количество раз. Также допустимо, если он может удалить только 1 символ из 1 индекса в строке, а остальные символы будут встречаться одинаковое количество раз. Для данной строки str определите, действительна ли она. Если да, верните «ДА», в противном случае верните «НЕТ».

Например, если str = "abc", строка действительна, потому что частота символов у всех одинакова. Если str = "abcc", строка также действительна, потому что мы можем удалить 1 "c" и оставить по одному символу каждого символа в строке. Однако, если str = "abccc", строка недействительна, поскольку удаление одного символа не приводит к одинаковой частоте символов.

Пример:

isValid("aabbcd") ➞ "NO"

// We would need to remove two characters, both c and d -> aabb or a and b -> abcd, to make it valid.

// We are limited to removing only one character, so it is invalid.

isValid("aabbccddeefghi") ➞ "NO"

// Frequency counts for the letters are as follows:

// {"a": 2, "b": 2, "c": 2, "d": 2, "e": 2, "f": 1, "g": 1, "h": 1, "i": 1}

// There are two ways to make the valid string:

// Remove 4 characters with a frequency of 1: {f, g, h, i}.

// Remove 5 characters of frequency 2: {a, b, c, d, e}.

// Neither of these is an option.

isValid("abcdefghhgfedecba") ➞ "YES"

// All characters occur twice except for e which occurs 3 times.

// We can delete one instance of e to have a valid string.

1. Создайте функцию, которая будет находить наибольшую общую подпоследовательность (LCS) для двух строк. LCS – это самая длинная последовательность символов, которая встречается как подпоследовательность в обеих строках. Эта задача требует понимания алгоритма динамического программирования для нахождения наибольшей общей подпоследовательности и его эффективной реализации.

Пример:

findLCS("abcd", "bd") ➞ "bd"

findLCS("aggtab", "gxtxamb") ➞ "gtab"