**4/6**

1. Напишите рекурсивную функцию, которая принимает строку и удаляет из неё повторяющиеся символы. Функция должна вернуть строку, в которой каждый символ встречается только один раз.

Пример:

nonRepeatable("abracadabra") ➞ " abrcd"

nonRepeatable("paparazzi") ➞ " parzi"

1. Напишите функцию, которая генерирует все возможные правильные комбинации пар скобок для заданного числа n.

Пример:

generateBrackets(1) ➞ ["()"]

generateBrackets(2) ➞ ["(())", "()()"]

generateBrackets(3) ➞ ["((()))", "(()())", "(())()", "()(())", "()()()"]

1. Напишите функцию, которая генерирует все возможные бинарные комбинации длины n, в которых не может быть соседствующих нулей или единиц.

Пример:

binarySystem(3) ➞ ["010", "011", "101", "110", "111"]

binarySystem(4) ➞ ["0101", "0110", "0111", "1010", "1011", "1101", "1110", "1111"]

1. Реализуйте функцию, которая принимает строку и возвращает длину самого длинного последовательного ряда в этом массиве. Последовательный ряд – это список соседних элементов, идущих подряд в алфавитном порядке, который может быть как увеличивающимся, так и уменьшающимся.

Пример:

alphabeticRow("abcdjuwx") ➞ "abcd"

// два последовательных ряда: "abcd", "uwx"; самый длинный: "abcd"

alphabeticRow("klmabzyxw") ➞ "zyxw"

1. Напишите функцию, которая принимает строку и подсчитывает количество идущих подряд символов, заменяя соответствующим числом повторяющиеся символы. Отсортируйте строку по возрастанию длины буквенного паттерна.

Пример:

("aaabbcdd") ➞ "c1b2d2a3"

("vvvvaajaaaaa") ➞ "j1a2v4a5"

1. Напишите функцию, принимающую положительное целое число в строковом формате, не превышающее 1000, и возвращающую его целочисленное представление.

Пример:

convertToNum("eight") ➞ 8

convertToNum("five hundred sixty seven") ➞ 567

convertToNum("thirty one") ➞ 31

1. Напишите функцию, принимающую строку цифр, выполняющую поиск подстроки максимальной длины с уникальными элементами. Если найдено несколько подстрок одинаковой длины, верните первую.

Пример:

uniqueSubstring("123412324") ➞ "1234"

uniqueSubstring("111111") ➞ "1"

uniqueSubstring("77897898") ➞ "789"

1. Напишите функцию поисковик наименьшего матричного пути. На вход поступает двумерный массив, размера n x n, ваша задача найти путь с минимальной суммой чисел, передвигаясь только вправо или вниз.

Пример:

shortestWay(

[[1, 3, 1],

[1, 5, 1],

[4, 2, 1]]) ➞ 7

// 1+3+1+1+1=7

shortestWay(

[[2, 7, 3],

[1, 4, 8],

[4, 5, 9]]) ➞ 21

1. Создайте функцию, принимающую строку, содержащую числа внутри слов. Эти числа представляют расположение слова для новой строящейся строки.

Пример:

numericOrder("t3o the5m 1One all6 r4ule ri2ng") ➞ " One ring to rule them all"

numericOrder("re6sponsibility Wit1h gr5eat power3 4comes g2reat") ➞ " With great power comes great responsibility"

1. Напишите функцию, принимающую два числа, которая делает второе число максимально возможным за счет замены своих элементов элементами первого числа. Брать цифру можно только один раз.

Пример:

switchNums(519, 723) ➞ 953

switchNums(491, 3912) ➞ 9942

switchNums(6274, 71259) ➞ 77659