**1/6**

1. Создайте функцию, которая принимает целое число галлонов и преобразует его в литры.

Пример:

convert(5) ➞ 18.925

convert(3) ➞ 11.355

convert(8) ➞ 30.28

1. Вы пишете программу для квази-фитнес-приложения и хотите создать функцию для расчета калорий, сожженных пользователем во время тренировки. Функция должна принимать время тренировки в минутах и интенсивность, где 1 – низкая интенсивность, 2 – средняя, 3 – высокая, а затем вычислять количество сожженных калорий на основе этой информации.

Пример:

fitCalc(15, 1) ➞ 15

fitCalc(24, 2) ➞ 48

fitCalc(41, 3) ➞ 123

1. В этой задаче вы управляете складом, где хранятся товары трех типов:

- Коробки содержат по 20 товаров в каждой.

- Мешки содержат по 50 товаров в каждом.

- Бочки содержат по 100 товаров в каждой.

Вам предоставили информацию о количестве каждого типа емкостей на складе, и вам нужно создать функцию, которая вернет общее количество товаров на складе, учитывая объекты хранения разных типов.

Пример:

containers(3, 4, 2) ➞ 460

containers(5, 0, 2) ➞ 300

containers(4, 1, 4) ➞ 530

1. Создайте функцию, которая принимает 3 числа: X, Y и Z. Эти числа представляют длины сторон треугольника. Функция должна вернуть тип треугольника на основе данных сторон: "равносторонний" (если все стороны равны), "равнобедренный" (если две стороны равны), "разносторонний" (если все стороны разные) или "не является треугольником" (если невозможно построить треугольник с заданными сторонами).

Пример:

triangleType(5, 5, 5) ➞ isosceles

triangleType(5, 4, 5) ➞ equilateral

triangleType(3, 4, 5) ➞ different-sided

triangleType(5, 1, 1) ➞ not a triangle

1. В Java есть вариация условного оператора – тернарный оператор "? :", принимающий три операнда и возвращающий один из них на основе значения условия. Он имеет следующую структуру:

условие ? выражение1 : выражение2

Ваша задача создать функцию, которая принимает два числа a и b, а затем с помощью тернарного оператора определяет, какое из чисел больше, и возвращает большее число.

Пример:

ternaryEvaluation(8, 4) ➞ 8

ternaryEvaluation(1, 11) ➞ 11

ternaryEvaluation(5, 9) ➞ 9

1. У меня есть ограниченное количество ткани определенной длины, и я хочу сшить как можно больше пододеяльников. Создайте функцию, которая будет принимать длину ткани (в метрах) и размер одной детали (ширина и длина в метрах), а затем возвращать количество пододеяльников, которые я смогу сшить, прежде чем кончится рулон.

n \* 2 – это количество квадратных метров имеющейся ткани,

w и h – это длина и ширина одной детали в метрах

Пример:

howManyItems(22, 1.4, 2) ➞ 3

howManyItems(45, 1.8, 1.9) ➞ 6

howManyItems(100, 2, 2) ➞ 12

Примечание:

- Не считайте пододеяльник, если на него не хватило ткани в рулоне

1. Напишите функцию, вычисляющую факториал выбранного числа.

Пример:

factorial(3) ➞ 6

factorial(5) ➞ 120

factorial(7) ➞ 5040

1. Создайте функцию, которая находит наибольший общий делитель двух чисел.

Пример:

gcd(48, 18) ➞ 6

gcd(52, 8) ➞ 4

gcd(259, 28) ➞ 1

1. Создайте функцию, которая принимает количество билетов на концерт, проданных через веб-сервис, и стоимость одного билета с учетом фиксированной комиссии. Функция должна вернуть общую выручку от продажи билетов.

Пример:

ticketSaler(70, 1500) ➞ 75600

ticketSaler(24, 950) ➞ 16416

ticketSaler(53, 1250) ➞ 47700

1. Создайте функцию, которая принимает целое число студентов и количество парт в аудитории. Функция должна определить, сколько столов не хватает для размещения всех студентов, если за одним столом помещается два студента.

Пример:

tables(5, 2) ➞ 1

tables(31, 20) ➞ 0

tables(123, 58) ➞ 4

**2/6**

1. Создайте функцию, которая определяет, есть ли в строке повторяющиеся символы.

Пример:

duplicateChars("Donald") ➞ true

duplicateChars("orange") ➞ false

1. Создайте метод, который принимает строку (фамилию и имя человека) и возвращает строку с инициалами без пробелов.

Пример:

getInitials("Ryan Gosling") ➞ "RG"

getInitials("Barack Obama") ➞ "BA"

1. Создайте функцию, которая принимает массив и возвращает разницу между суммой четных и нечетных.

Пример:

differenceEvenOdd([44, 32, 86, 19]) ➞ 143

differenceEvenOdd([22, 50, 16, 63, 31, 55]) ➞ 61

1. Создайте функцию, которая принимает массив и возвращает true, если в массиве есть хотя бы один элемент, который равен среднему арифметическому всех элементов массива, и false в противном случае.

Пример:

equalToAvg([1, 2, 3, 4, 5]) ➞ true

equalToAvg ([1, 2, 3, 4, 6]) ➞ false

1. Создайте метод, который берет массив целых чисел и возвращает массив, в котором каждое целое число умножено на индекс этого числа в массиве.

Пример:

indexMult([1, 2, 3]) ➞ [0, 2, 6]

indexMult([3, 3, -2, 408, 3, 31]) ➞ [0, 3, -4, 1224, 12, 155]

1. Создайте метод, который принимает строку в качестве аргумента и возвращает ее в обратном порядке.

Пример:

reverse("Hello World") ➞ "dlroW olleH"

reverse("The quick brown fox.") ➞ ".xof nworb kciuq ehT"

1. Создайте функцию, которая при заданном числе возвращает соответствующее число Трибоначчи. Последовательность Трибоначчи начинается с элементов «0, 0, 1».

Пример:

Tribonacci(7) ➞ 7

Tribonacci(11) ➞ 81

1. Хэш-суммы в системе контроля версий (например, Git) выглядят как уникальная строка из символов (от *a* до *f*) и цифр (от 0 до 9) длиной в 40 элементов. В Git используется SHA-1 хэш-функция для создания хэшей коммитов.

Создайте функцию, генерирующую квази-хэш заданной пользователем длины.

Пример:

pseudoHash(5) ➞ "04bf2"

pseudoHash(10) ➞ "2d9c45e1f3"

pseudoHash(0) ➞ ""

1. Напишите функцию, которая находит слово "help" в данной строке-стенограмме автоматизированного телефонного диспетчера службы спасения. Ответьте "Вызов сотрудника", если слово найдено, в противном случае – "Продолжайте ожидание".

Пример:

botHelper("Hello, I’m under the water, please help me") ➞ "Calling for a staff member"

botHelper("Two pepperoni pizzas please") ➞ "Keep waiting"

Примечание:

Строка "help" может появляться в разных случаях символов (например, в верхнем, нижнем регистре).

1. Создайте функцию, которая принимает две строки и определяет, являются ли они анаграммами.

Пример:

isAnagram("listen", "silent") ➞ true

isAnagram("eleven plus two", "twelve plus one") ➞ true

isAnagram("hello", "world") ➞ false

**3/6**

1. Создайте функцию, которая принимает строку и заменяет все гласные буквы на символ «\*».

Пример:

replaceVovels("apple") ➞ "\*ppl\*"

duplicateChars("Even if you did this task not by yourself, you have to understand every single line of code.") ➞ "\*v\*n \*f \*\*\* d\*d th\*s t\*sk n\*t b\* \*\*\*rs\*lf, \*\*\* h\*v\* t\* \*nd\*rst\*nd \*v\*r\* s\*ngl\* l\*n\* \*f c\*d\*."

1. Напишите функцию, которая принимает строку и заменяет две идущие подряд буквы по шаблону «Double\*».

Пример:

stringTransform("hello") ➞ "heDoubleLo"

stringTransform("bookkeeper") ➞ "bDoubleODoubleKDoubleEper"

1. Помогите ребенку разобраться с игрушкой на развитие - поместится ли параллелепипед в коробку с отверстиями определенных параметров. Напишите функцию, которая принимает три измерения игрушечного блока: высоту(a), ширину(b) и глубину(c) и возвращает true, если этот блок может поместиться в отверстие с шириной(w) и высотой(h).

Пример:

doesBlockkFit(1, 3, 5, 4, 5) ➞ true

doesBlockFit(1, 8, 1, 1, 1) ➞ true

doesBlockFit(1, 2, 2, 1, 1) ➞ false

Примечание:

- Вы можете повернуть блок любой стороной к отверстию.

- Мы предполагаем, что блок подходит, если его размеры равны размерам отверстия, а не строго меньше.

- Блок можно класть только под прямым углом к поверхности.

1. Создайте функцию, которая принимает число в качестве входных данных и возвращает true, если сумма квадратов его цифр имеет ту же четность, что и само число. В противном случае верните false.

Пример:

numCheck(243) ➞ true

// 243 нечетное, как и 29 (2^2 = 4, 4^2 = 16, 3^2 = 9, 4+16+9 = 29)

numCheck(52) ➞ false

// 52 четное, но 29 – нет (5^2=25, 2^2=4, 25 + 4 = 29)

1. Создайте метод, который берет массив целых чисел-коэффициентов и возвращает количество целочисленных корней квадратного уравнения.

Пример:

countRoots([1, -3, 2]) ➞ 2

countRoots([2, 5, 2]) ➞ 1

countRoots([1, -6, 9]) ➞ 1

1. Создайте метод, который принимает двумерный массив, представляющий информацию о продажах разных товаров в различных магазинах, и возвращает товары, которые были проданы в каждом из магазинов.

Пример:

salesData([

["Apple", "Shop1", "Shop2", "Shop3", "Shop4"],

["Banana", "Shop2", "Shop3", "Shop4"],

["Orange", "Shop1", "Shop3", "Shop4"],

["Pear", "Shop2", "Shop4"]

]) ➞ ["Apple"]

salesData([

["Fridge", "Shop2", "Shop3"],

["Microwave", "Shop1", "Shop2", "Shop3", "Shop4"],

["Laptop", "Shop3", "Shop4"],

["Phone", "Shop1", "Shop2", "Shop3", "Shop4"]

]) ➞ ["Microwave", "Phone"]

1. Создайте функцию, которая определяет, можно ли разбить заданное предложение на слова так, чтобы каждое слово начиналось с последней буквы предыдущего слова.

Пример:

validSplit("apple eagle egg goat") ➞ true

validSplit("cat dog goose fish") ➞ false

1. Напишите метод, который определяет, является ли заданный массив «волнообразным». Последовательность чисел считается волнообразной, если разница между соседними элементами чередуется между убыванием и возрастанием.

Пример:

waveForm([3, 1, 4, 2, 7, 5]) ➞ true

последовательность начинается с убывания (3, 1), сменяющегося на возрастание (1, 4) и т.д.

waveForm([1, 2, 3, 4, 5]) ➞ false

waveForm([1, 2, -6, 10, 3]) ➞ true

1. Напишите функцию, которая находит наиболее часто встречающуюся гласную в предложении.

Пример:

commonVovel("Hello world") ➞ "o"

commonVovel("Actions speak louder than words.") ➞ "a"

1. Создайте функцию, которая принимает n целочисленных массивов длины n, а затем изменяет каждый n-ый элемент n-го массива на среднее арифметическое элементов n-го столбца остальных массивов.

Пример:

dataScience([

[1, 2, 3, 4, 5],

[6, 7, 8, 9, 10],

[5, 5, 5, 5, 5],

[7, 4, 3, 14, 2],

[1, 0, 11, 10, 1]

]) ➞

[[**5**, 2, 3, 4, 5],

[6, **3**, 8, 29, 10],

[5, 5, **6**, 5, 35],

[7, 4, 3, **12**, 2],

[1, 0, 11, 10, **13**]]

dataScience([

[6, 4, 19, 0, 0],

[81, 25, 3, 1, 17],

[48, 12, 60, 32, 14],

[91, 47, 16, 65, 217],

[5, 73, 0, 4, 21]

]) ➞

[[**56**, 4, 19, 0, 0],

[81, **34**, 3, 1, 17],

[48, 12, **10**, 32, 14],

[91, 47, 16, **9**, 217],

[5, 73, 0, 4, **62**]]

**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