JS Elementary Tasks

# Правила оформления заданий

* Точка входа в приложение – файл ***main.js***.
* Все реализации находятся внутри папки ***src***
* Файлы каждого задания именуются как ***task + номеру задания.js***. В итоге структура каталога со всеми заданиями будет выглядеть следующим образом:

***main.js***

***src/***

***task1.js***

***task2.js***

***task3.js***

***task….***

# Общие требования

1. При передаче некорректных параметров на исполнение приложение не должно завершать работу сбоем.
2. Запуск с некорректными параметрами (без параметров) возвращает инструкции по использованию программы. Инструкция должна иметь формат:

*{status: ‘failed’, reason: ‘ инструкция ‘}*

1. Параметры передаются в порядке, приведённом в описании задания.

# Задания

## Шахматная доска

Вывести шахматную доску с заданными размерами высоты и ширины, по принципу:

\* \* \* \* \* \*

\* \* \* \* \* \*

\* \* \* \* \* \*

\* \* \* \* \* \*

*Входные параметры*: длина, ширина, символ для отображения.

*Выход*: строка с представлением шахматной доски

## Анализ конвертов

Есть два конверта со сторонами (a,b) и (c,d). Требуется определить, можно ли один конверт вложить в другой. Программа должна обрабатывать ввод чисел с плавающей точкой.

*Входные параметры*: объекты *конверт1* и *конверт2*

*Выход*: номер конверта, если вложение возможно, 0 – если вложение невозможно.

## Сортировка треугольников

Вывести треугольники в порядке убывания их площади.

*Входные параметры*: массив объектов *треугольник*

*Выход*: упорядоченный массив имён треугольников

***Примечание:***

• Расчёт площади треугольника должен производится по формуле Герона.

• Каждый треугольник определяется именами вершин и длинами его сторон.

• Приложение должно обрабатывать ввод чисел с плавающей точкой:

*{*

*vertices: ‘ABC’,*

*a: 10,*

*b: 20,*

*c: 22.36*

*}*

## Палиндром

Проверить является ли число или его часть палиндромом. Например, число 1234437 не является палиндромом, но является палиндромом его часть 3443. Числа меньшие, чем 10 палиндромом не считать.

*Входные параметры*: число

*Выход*: извлечённый из числа палиндром либо 0, если извлечение не удалось.

## Счастливые билеты

Есть 2 способа подсчёта счастливых билетов:

*1. Простой — если на билете напечатано шестизначное число, и сумма первых трёх цифр равна сумме последних трёх, то этот билет считается счастливым.  
2. Сложный — если сумма чётных цифр билета равна сумме нечётных цифр билета, то билет считается счастливым.*

Определить программно какой вариант подсчёта счастливых билетов даст их большее количество на заданном интервале.

*Входные параметры*: объект *context* с полями *min* и *max*

*Выход*: объект с информацией о победившем методе и количестве счастливых билетов для каждого способа подсчёта.

## Числовая последовательность

Вывести через запятую ряд длиной n, состоящий из натуральных чисел, квадрат которых не меньше заданного m.

*Входные параметры*: длина и значение минимального квадрата

*Выход*: строка с рядом чисел

## Ряд Фибоначчи для диапазона

Вывести все числа Фибоначчи, которые удовлетворяют переданному в функцию ограничению: находятся в указанном диапазоне, либо имеют указанную длину.

*Входные параметры*: объект *context* с полями *min* и *max,* либо с полем *length*

*Выход*: массив чисел