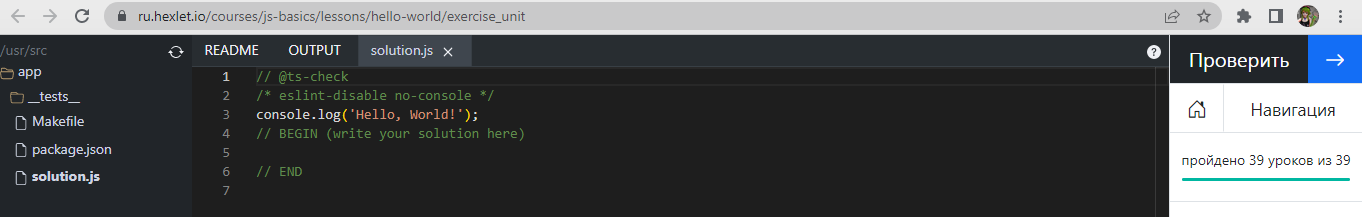
1. 1 урок не имеет практического упражнения.

2. Наберите в редакторе код из примера ниже символ в символ и нажмите «Проверить».

console.log('Hello, World!');

Решение:



3. Выведите на экран друг за другом три имени: Robert, Stannis, Renly. В результате на экране должно отобразиться:

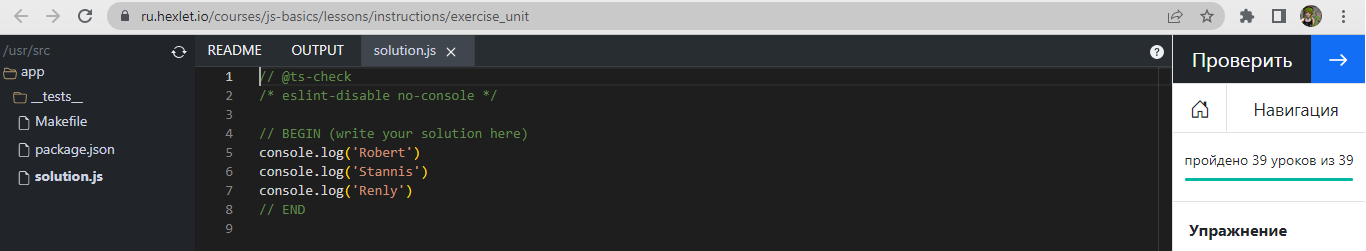
Robert

Stannis

Renly

Для каждого имени используйте свой собственный вызов console.log(). Номера строк выводятся автоматически, их не нужно печатать. Нужно выводить только строки с именами.

Решение:



4. Напишите программу, которая считает и выводит на экран последовательно (по одному значению в каждой строке) значения следующих математических выражений:

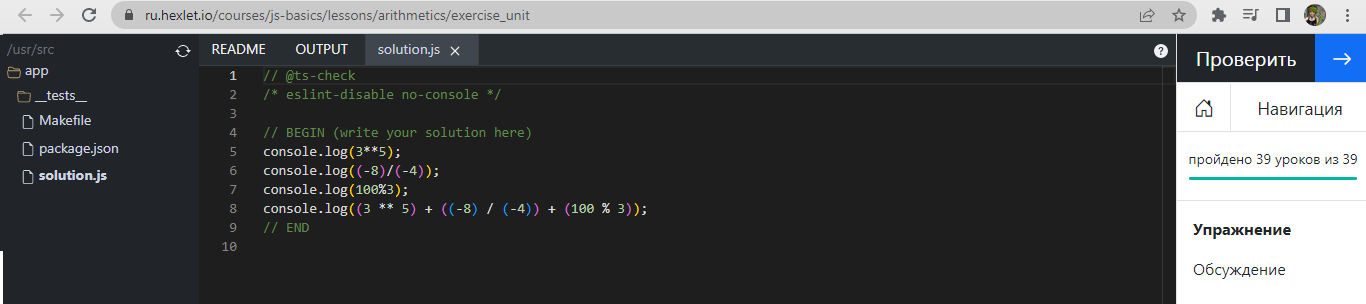
3 в степени 5

-8 разделить на -4

остаток от деления 100 на 3

сумму трёх предыдущих выражений

Решение:

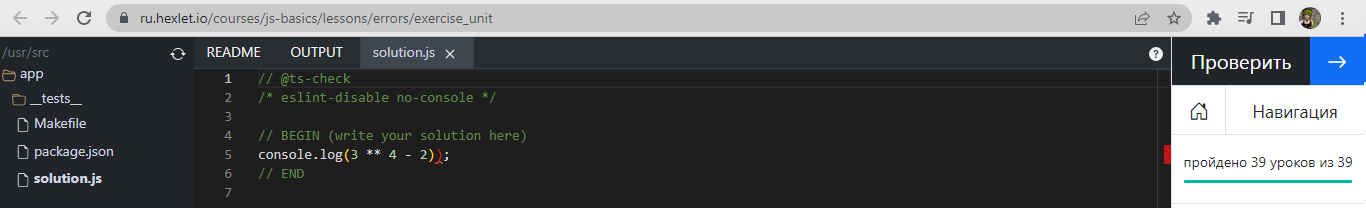


5. Проанализируйте этот код и скопируйте его в редактор:

console.log(3 \*\* (4 - 2));

Внесите в код такие изменения, которые приведут к синтаксической ошибке.

Решение:



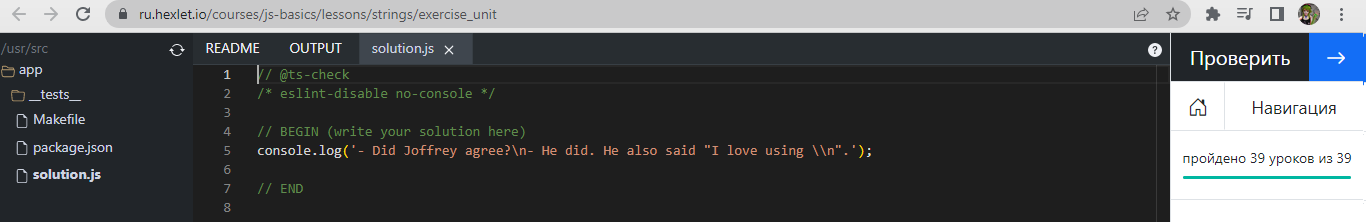
6. Напишите программу, которая выводит на экран:

- Did Joffrey agree?

- He did. He also said "I love using \n".

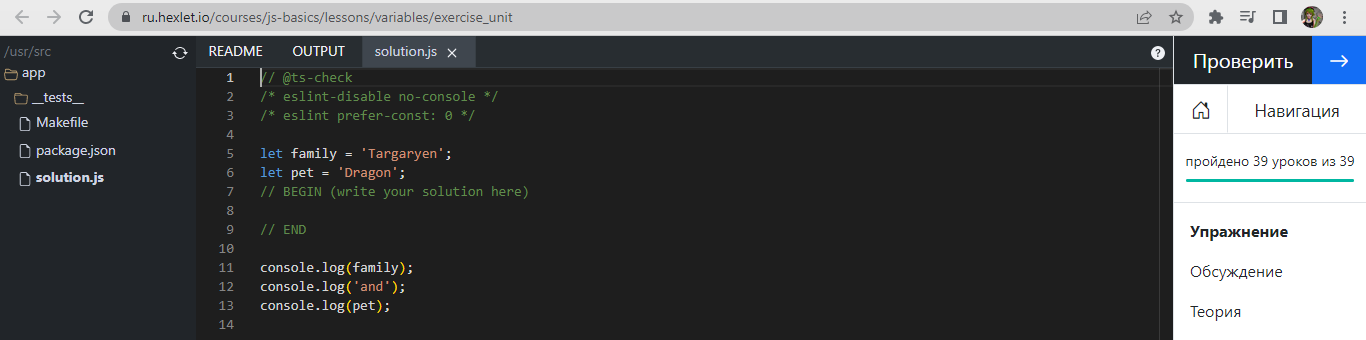
При этом программа использует только один console.log(), но результат на экране должен выглядеть в точности, как показано выше.

Решение:



7. Найдите в программе необъявленную переменную и объявите её, присвоив ей значение Dragon.

Решение:



8. Напишите программу, которая берет исходное количество евро, записанное в константу eurosCount, переводит евро в доллары и выводит на экран. Затем полученное значение переводит в рубли и выводит на новой строчке.

Пример вывода для 100 евро:

125

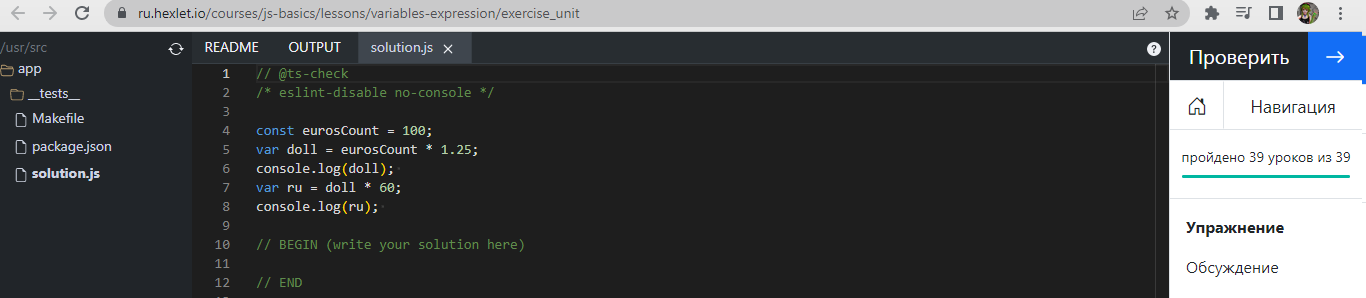
7500

Считаем, что:

1 евро = 1.25 долларов

1 доллар = 60 рублей

Решение:



9. solution.js

Вы столкнулись с таким кодом, который выводит на экран общее количество комнат во владении нынешнего короля:

const king = 'King Balon the 6th';

console.log(king + ' has ' + (6 \* 17) + ' rooms.');

Как видите, это магические числа: непонятно, что такое 6 и что такое 17. Можно догадаться, если знать историю королевской семьи: каждый новый король получает в наследство все замки от предков и строит новый замок — точную копию родительского. Эта странная династия просто плодит одинаковые замки…

Избавьтесь от магических чисел, создав новые переменные, а затем выведите текст на экран:

King Balon the 6th has 102 rooms.

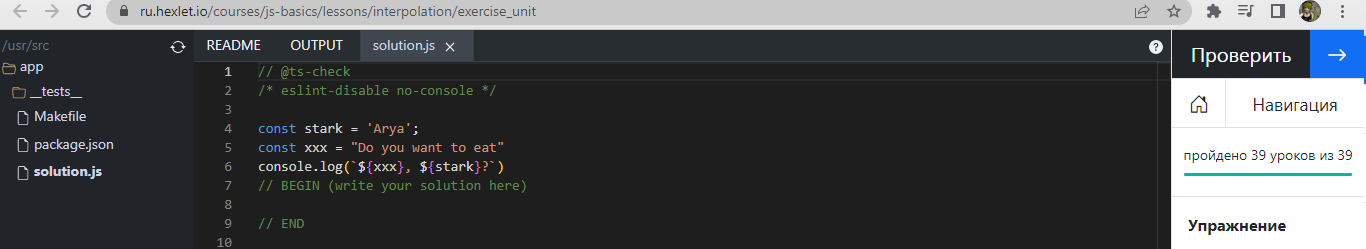
Названия переменных должны передавать смысл чисел, но должны при этом оставаться достаточно короткими и ёмкими для комфортного чтения.

Решение:

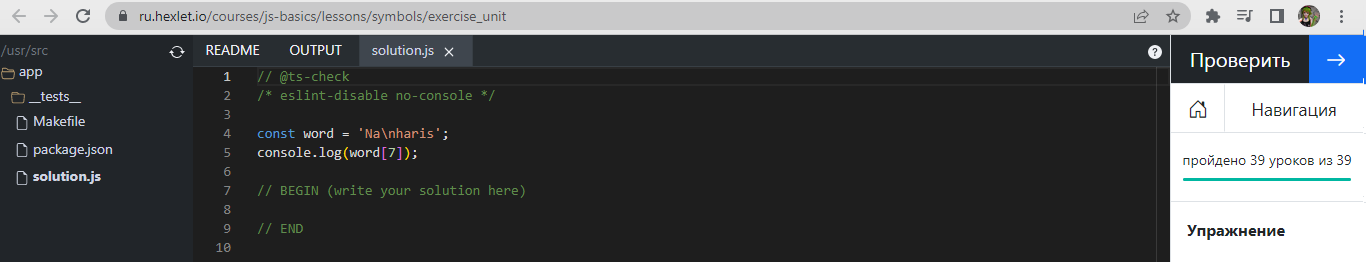


10. Выведите на экран строку Do you want to eat, <name>?, где вместо <name> должна использоваться константа stark. Вывод должен получиться таким:

Do you want to eat, Arya?

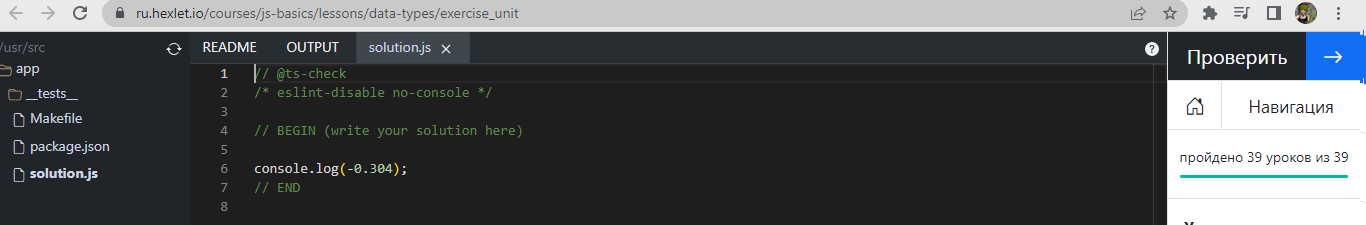
Решение:

11. Выведите на экран последний символ строки, находящейся в константе word.

Решение:

12. Выведите на экран число -0.304.

Решение:



13. Вам даны три переменные с фамилиями разных людей. Составьте и выведите на экран слово из символов в таком порядке:

Третий символ из первой строки

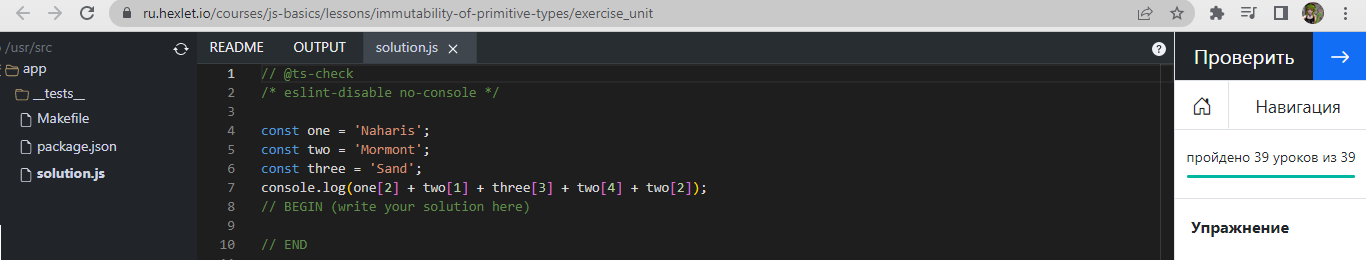
Второй символ из второй строки

Четвертый символ из третьей строки

Пятый символ из второй строки

Третий символ из второй строки

Попробуйте использовать интерполяцию. Внутри фигурных скобок можно размещать не только переменные, но и отдельные символы строки, извлечённые по индексу (с помощью квадратных скобок).

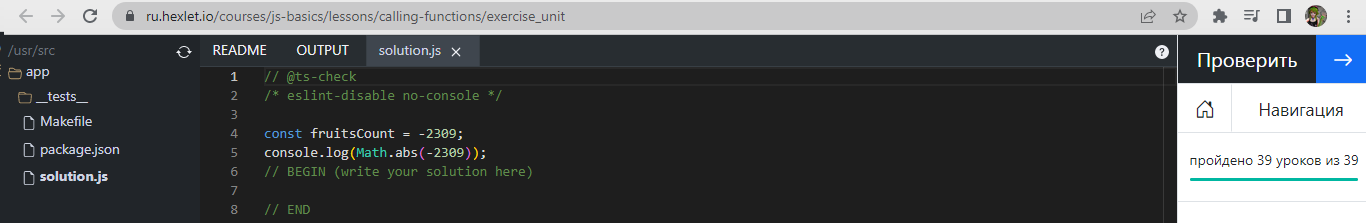
Решение:

14. Выведите на экран абсолютное значение числа, находящегося в константе fruitsCount, используя функцию Math.abs(). Эта функция возвращает абсолютное значение переданного числа. Абсолютное значения числа — это само это число без знака перед ним.

Math.abs(-3); // 3

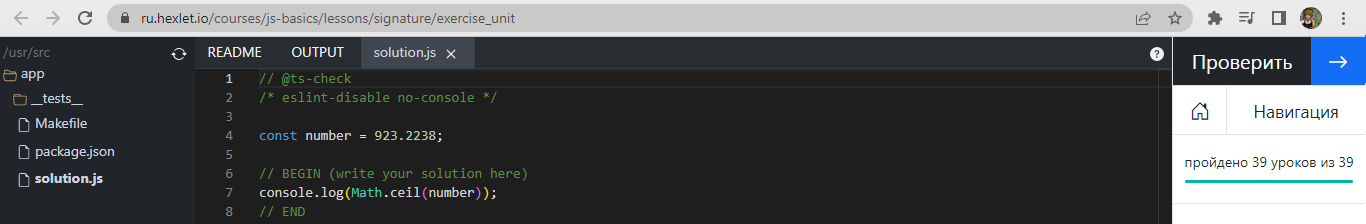
Math.abs(3); // 3

Math.abs(0); // 0

Решение:

15. Теперь ваша очередь посмотреть на сигнатуру функции в документации и разобраться, как её использовать. Можете читать документацию на русском языке, но программист должен уметь читать документацию на английском. Используйте словари или переводчики при необходимости. Лучше сразу привыкать и подтягивать навыки чтения на английском, иначе будут сложности в будущем. В Math есть функция ceil(). Изучите её документацию.

Напишите программу, которая использует функцию Math.ceil() с константой number и выводит результат на экран.

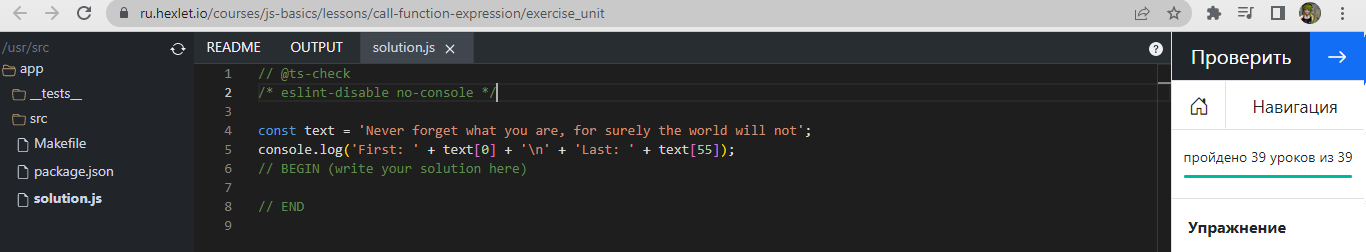
Решение:

16. Выведите на экран первую и последнюю буквы предложения, записанного в константу text, в следующем формате:

First: N

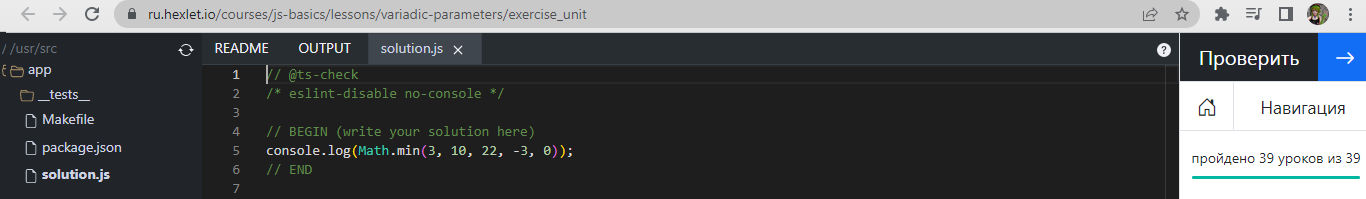
Last: t

Постарайтесь создать только одну константу, в которую сразу запишется нужный текст перед печатью на экран. В этом уроке мы отрабатываем умение собирать составное выражение.

Решение:

17. Посчитайте программно (а не в голове) минимальное число среди 3, 10, 22, -3, 0 — и выведите его на экран. Воспользуйтесь функцией Math.min(), которая работает аналогично Math.max().

Решение:

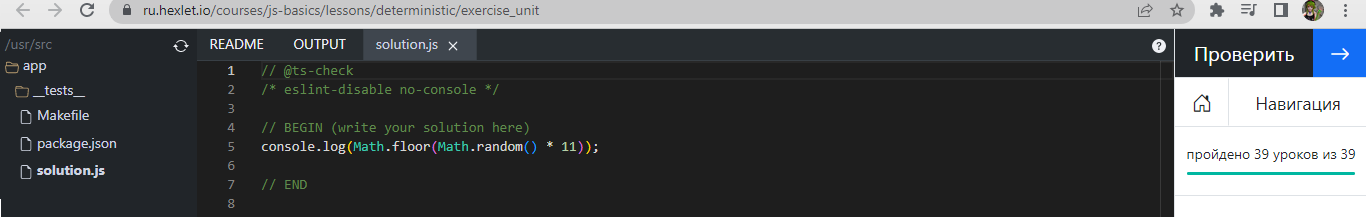


18. Функция Math.random() возвращает случайное число от 0 до 1 с большим количеством знаков после запятой. Но в реальных задачах бывает нужно получать случайные целые числа, например, в диапазоне от 0 до 10 включительно. Реализуйте код, который выводит на экран именно такое число. Для этой задачи вам понадобятся функции Math.random() и Math.floor()

Попробуйте решить это задание в одну строчку

Алгоритм

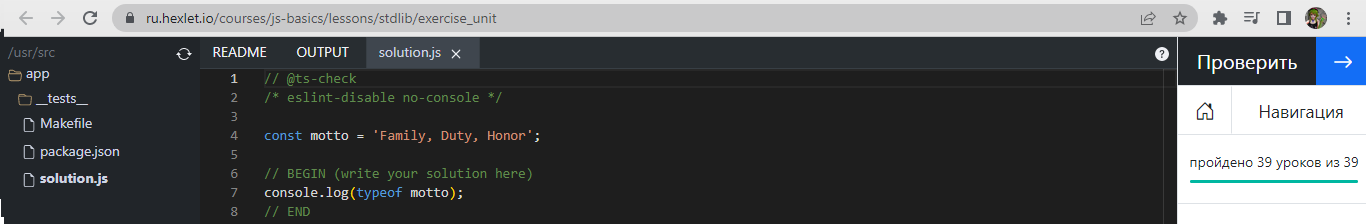
Так как Math.random() возвращает числа в диапазоне от 0 (включительно) до 1 (не включая саму единицу), то чтобы получить числа от 0 до 10 включительно, нам нужно выполнить умножение на 11. Затем получившиеся число округляется до ближайшего меньшего целого. И так мы получаем то, что нужно.

Решение:

19. Оператор typeof позволяет определить тип передаваемого операнда. Название типа возвращается в виде строки. Например, вызов typeof 'go go go' вернёт строку 'string' (number — число).

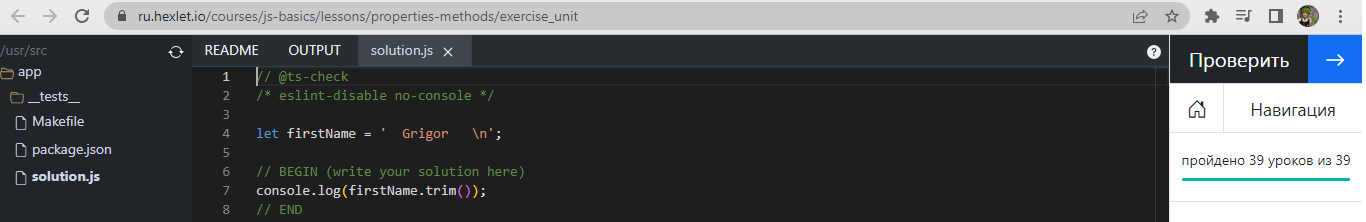
console.log(typeof 3); // => 'number'

Выведите на экран тип значения константы motto.

Решение:

20. Данные, вводимые пользователями, часто содержат лишние пробельные символы в конце или начале строки. Обычно их вырезают с помощью метода .trim(), например, было: ' hello\n ', стало: 'hello'.

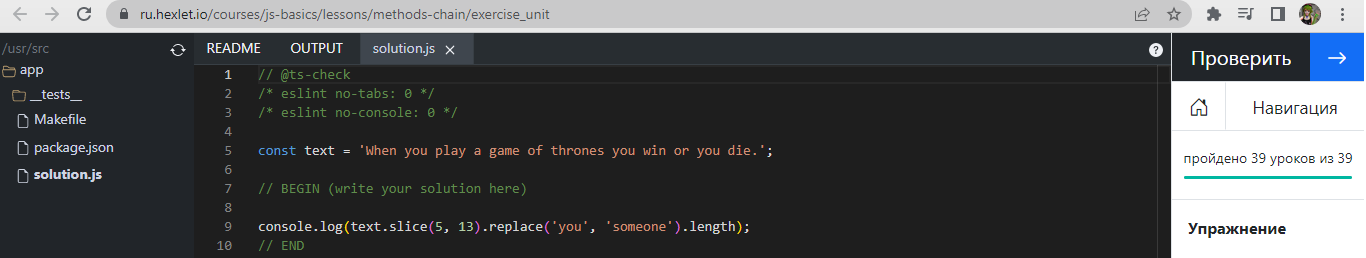
Обновите переменную firstName записав в неё то же самое значение, но обработанное методом .trim(). Распечатайте то, что получилось, на экран.

Решение:

21. С помощью метода .slice() получите часть предложения, записанного в константу text, c 6 по 13 символы включительно. В полученной подстроке с помощью метода .replace() замените слово you на someone и выведите на экран длину итоговой подстроки. Выполните эти методы подряд в цепочке без создания промежуточных переменных.

slice()

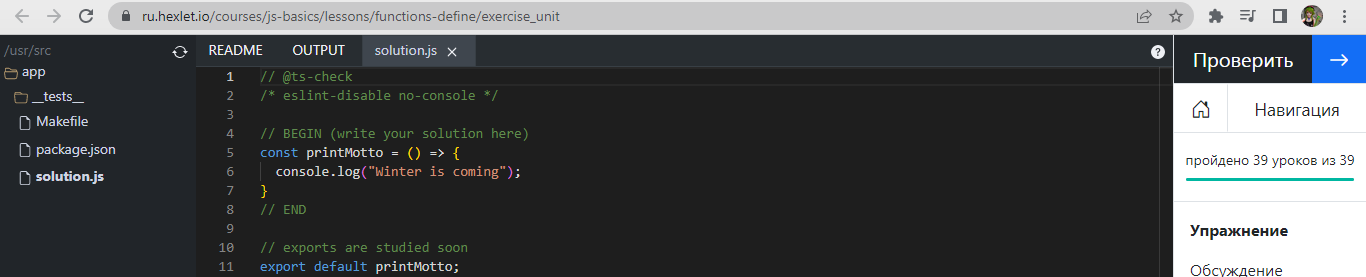
replace()

Решение:

22. Реализуйте функцию с именем printMotto(), которая выведет на экран фразу Winter is coming.

printMotto(); // => "Winter is coming"

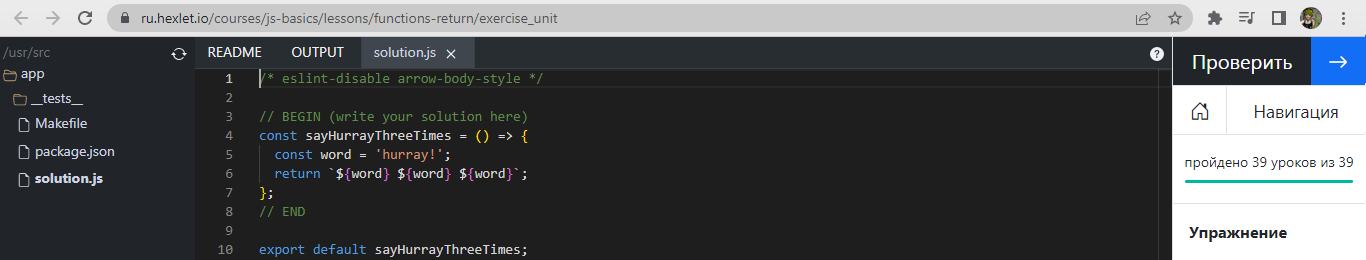
В задачах, в которых нужно реализовать функцию, эту функцию вызывать не нужно. Вызывать функцию будут автоматизированные тесты, которые проверяют его работоспособность. Пример с вызовом выше показан только для того, чтобы вы понимали, как ваша функция будет использоваться.

Решение:

23. Реализуйте функцию sayHurrayThreeTimes(), которая возвращает строку 'hurray! hurray! hurray!'.

const hurray = sayHurrayThreeTimes();

console.log(hurray); // => hurray! hurray! hurray!

Решение:

24. Реализуйте функцию truncate(), которая обрезает переданную строку до указанного количества символов, добавляет в конце многоточие и возвращает получившуюся строку. Подобная логика часто используется на сайтах, чтобы отобразить длинный текст в сокращенном виде.

Функция принимает два параметра:

Строка, которую нужно обрезать

Число символов, которые нужно оставить

// В функцию могут быть переданы в качестве параметров

// любая строка и любое количество символов

// Обрезаем текст 'hexlet', оставляя 2 символа

truncate('hexlet', 2); // "he..."

// Обрезаем текст 'it works!', оставляя 4 символа

truncate('it works!', 4); // 'it w...'

// Обрезаем текст 'hello', оставляя 3 символа

truncate('hello', 3); // 'hel...'

Подсказки

Вам нужно самостоятельно реализовать функцию truncate(), в JS такой функции не существует. Реализовать эту функцию можно различными способами, подскажем лишь один из них. Для решения этим способом вам понадобится взять подстроку из строки, переданной первым параметром в функцию. Используйте для этого метод slice(). Подумайте, исходя из задания, с какого индекса и по какой вам надо извлечь подстроку?

const word = 'welcome';

// Передавать параметры в метод можно через переменные

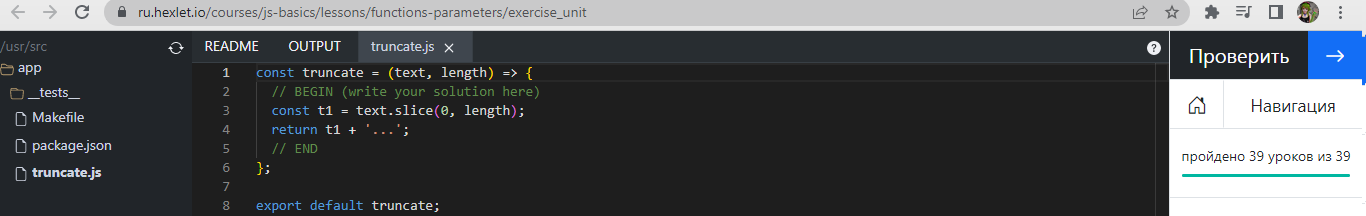
const index = 3;

word.slice(0, index); // "wel"

С точки зрения проверочной системы не имеет значения, каким из способов будет реализована функция truncate() внутри, главное – чтобы она выполняла поставленную задачу

Вам не нужно задавать конкретные значения для параметров, они будут переданы в функцию во время её тестирования.

Чтобы функция могла работать с любой строкой, используйте внутри функции переменные – параметры функции. После вызова функции эти переменные примут конкретные значения. Так функция сможет обрезать любую строку, которую мы передадим ей при вызове

Решение:

25. Реализуйте функцию getHiddenCard(), который принимает на вход номер кредитки (состоящий из 16 цифр) в виде строки и возвращает его скрытую версию, которая может использоваться на сайте для отображения. Если исходная карта имела номер 2034399002125581, то скрытая версия выглядит так \*\*\*\*5581. Другими словами, функция заменяет первые 12 символов, на звездочки. Количество звездочек регулируется вторым необязательным параметром. Значение по умолчанию — 4.

// Кредитка передается внутрь как строка

getHiddenCard('1234567812345678', 2); // "\*\*5678"

getHiddenCard('1234567812345678', 3); // "\*\*\*5678"

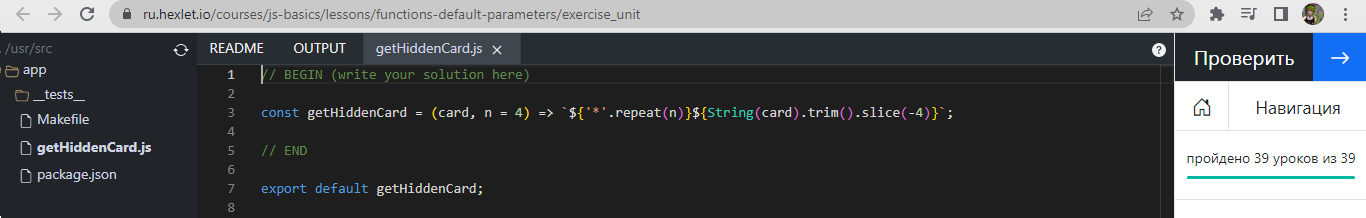
getHiddenCard('1234567812345678'); // "\*\*\*\*5678"

getHiddenCard('2034399002121100', 1); // "\*1100"

Для выполнения задания вам понадобится метод строки repeat(), который повторяет строку указанное количество раз

'+'.repeat(5); // "+++++"

'o'.repeat(3); // "ooo"

Решение:

26. Реализуйте функцию capitalize(), которая принимает непустую строку и приводит первую букву строки к верхнему регистру:

const name = 'arya';

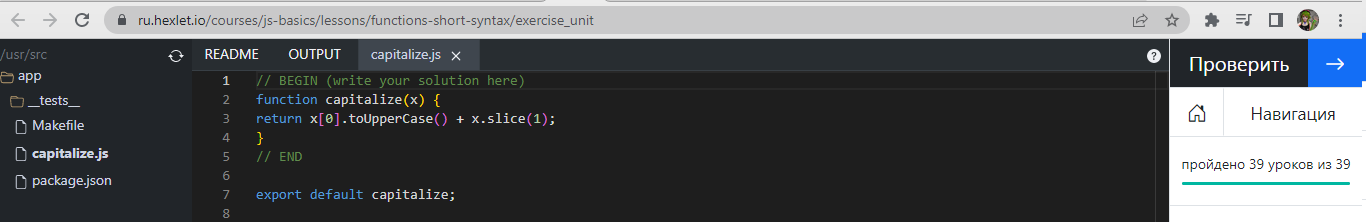
console.log(capitalize(name)); // => "Arya"

Чтобы получить подстроку (или символ) из строки, используйте метод slice():

'welcome'.slice(2, 5); // "lco"

Для приведения строки к верхнему регистру используйте метод toUpperCase()

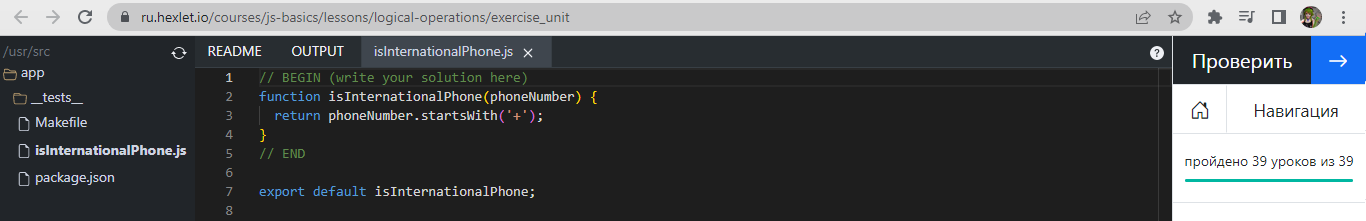
'welcome'.toUpperCase(); // "WELCOME"

Решение:

27. Реализуйте функцию isInternationalPhone(), которая проверяет формат указанного телефона. Если телефон начинается с +, значит это международный формат.

isInternationalPhone('89602223423'); // false

isInternationalPhone('+79602223423'); // true

Решение:

28. Реализуйте функцию isLeapYear(), которая определяет, является ли год високосным. Если год високосный, функция должна вернуть true, если нет – false. Год считается високосным, если он кратен (то есть делится без остатка) 400 или он одновременно кратен 4 и не кратен 100. Как видите, в определении уже заложена вся необходимая логика, осталось только переложить её на код:

isLeapYear(2018); // false

isLeapYear(2017); // false

isLeapYear(2016); // true

Кратность можно проверять так:

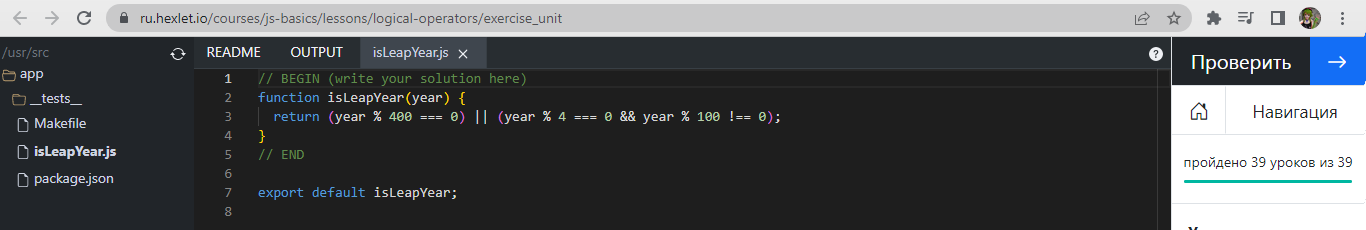
// % - возвращает остаток от деления левого операнда на правый

// Проверяем что number кратен 10

number % 10 === 0

// Проверяем что number не кратен 10

number % 10 !== 0

Решение:29. Реализуйте функцию getLetter(), которая извлекает из переданной строки указанный символ (по порядковому номеру, а не индексу) и возвращает его наружу. Если такого символа нет, то функция возвращает пустую строку.

Примеры вызова:

const name = 'Hexlet';

// Обычное обращение возвращает undefined

name[10]; // undefined

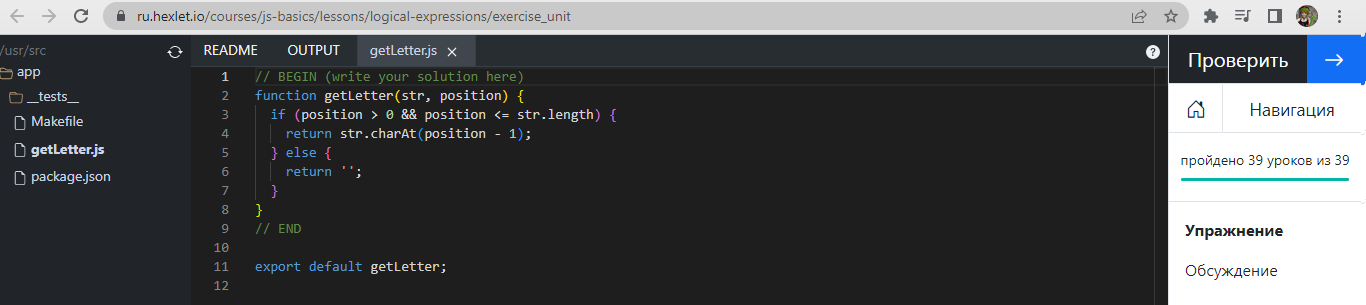
// 11 символ соответствует 10 индексу

getLetter(name, 11); // ''

getLetter(name, 1); // 'H'

getLetter(name, 0); // ''

getLetter(name, 6); // 't'

Решение:

30. Реализуйте функцию normalizeUrl(), которая выполняет так называемую нормализацию данных. Она принимает адрес сайта и возвращает его с https:// в начале.

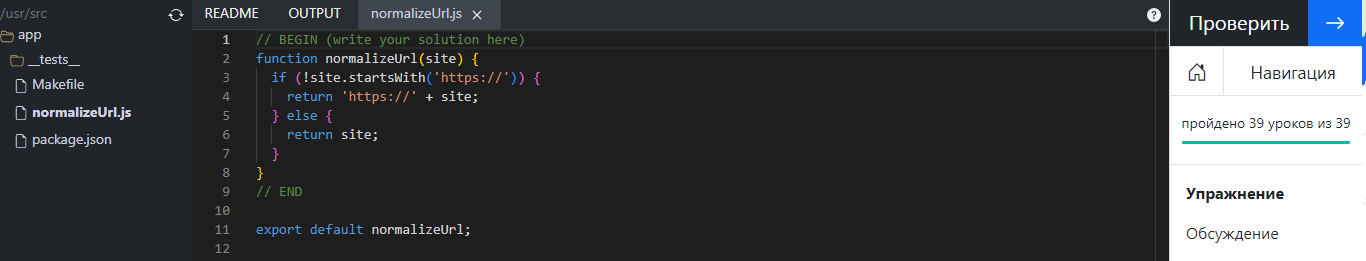
Функция принимает адреса в виде АДРЕС или https://АДРЕС, но всегда возвращает адрес в виде https://АДРЕС

Можно использовать метод startsWith(), чтобы проверить, начинается ли строка с префикса https://. А потом на основе этого добавлять или не добавлять https://.

Примеры вызова:

normalizeUrl("google.com"); // "https://google.com"

normalizeUrl("https://ai.fi"); // "https://ai.fi"

Решение:

31. Реализуйте функцию convertText(), которая принимает на вход строку и, если первая буква не заглавная, возвращает перевернутый вариант исходной строки. Если первая буква заглавная, то строка возвращается без изменений. Если на вход передана пустая строка, функция должна вернуть пустую строку.

Примеры вызова:

convertText('Hello'); // 'Hello'

convertText('hello'); // 'olleh'

// Не забудьте учесть пустую строку

convertText(''); // ''

Перевернуть строчку можно используя функцию reverse(). В качестве аргумента в неё нужно передать строку, которую мы хотим перевернуть:

const result = reverse('Hello!');

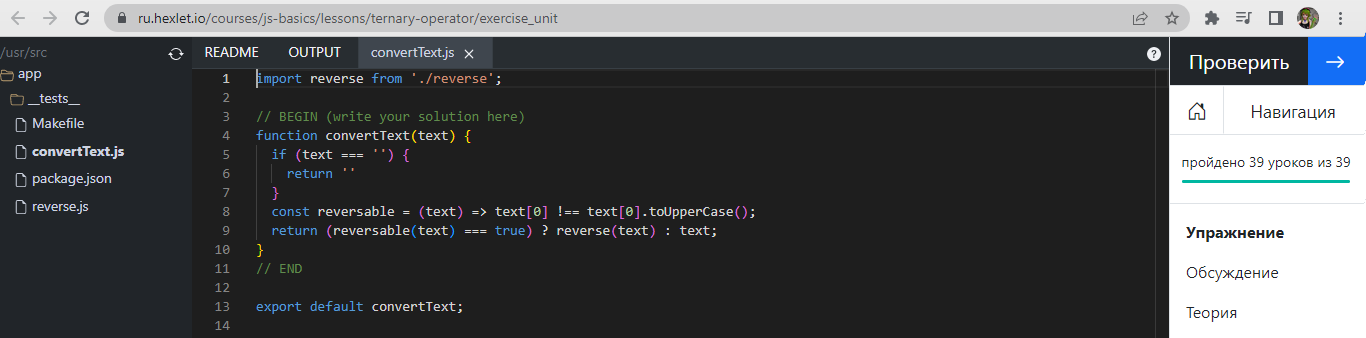
console.log(result); // => '!olleH'

Есть разные подходы к решению этой задачи. Возможно, вам пригодится метод toUpperCase() и возможность получения символа из строки (например, str[0]).

Попробуйте написать два варианта функции: с обычным if-else, и с тернарным оператором.

Подсказки

Подумайте какую проверку нужно написать первым делом: проверку заглавности буквы или является ли пустой строка. Что первичнее?

Решение:

32. Реализуйте функцию getNumberExplanation(), которая принимает на вход число и возвращает объяснение этого числа. Если для числа нет объяснения, то возвращается just a number. Объяснения есть только для следующих чисел:

\* 666 - devil number

\* 42 - answer for everything

\* 7 - prime number

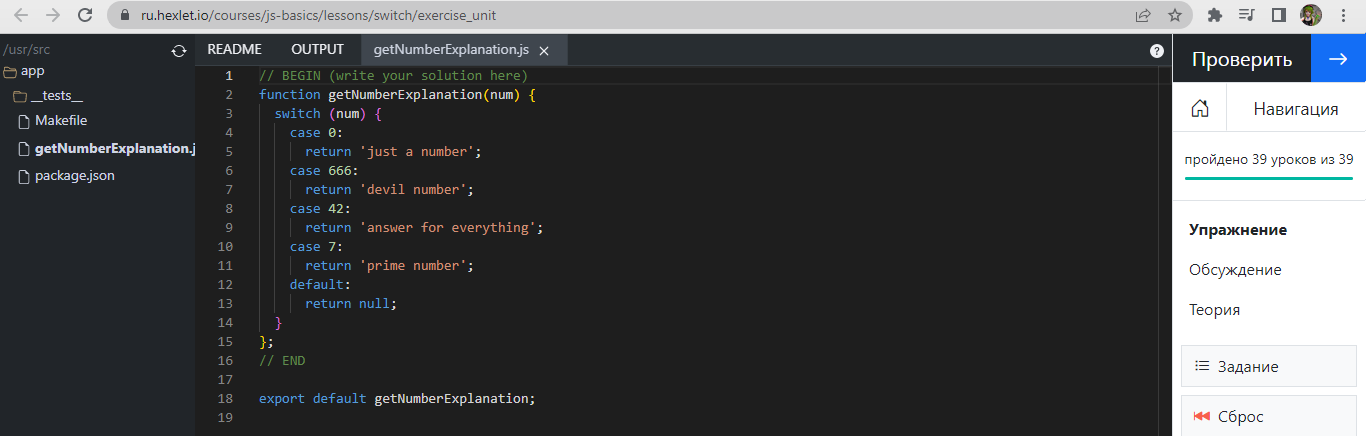
Примеры вызова функции:

getNumberExplanation(8); // just a number

getNumberExplanation(666); // devil number

getNumberExplanation(42); // answer for everything

getNumberExplanation(7); // prime number

Решение:

33. Модифицируйте функцию printNumbers() так, чтобы она выводила числа в обратном порядке. Для этого нужно идти от верхней границы к нижней. То есть счётчик должен быть инициализирован максимальным значением, а в теле цикла его нужно уменьшать до нижней границы.

printNumbers(4);

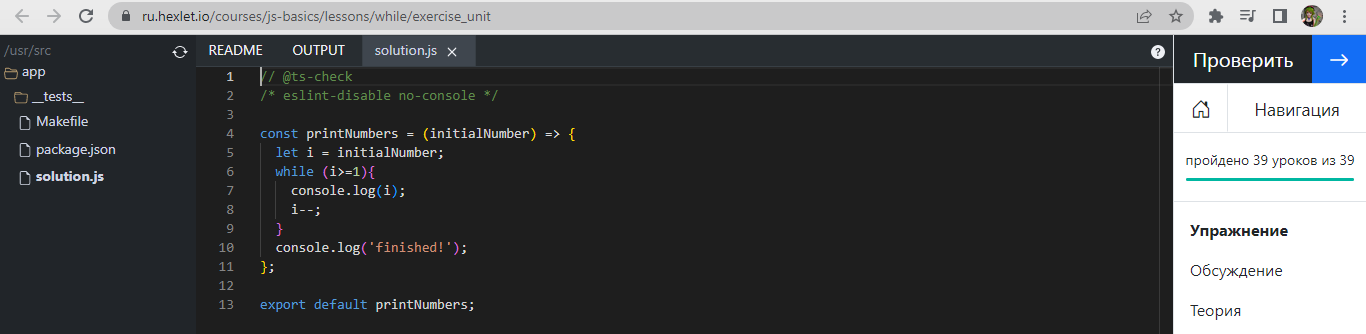
4

3

2

1

finished!

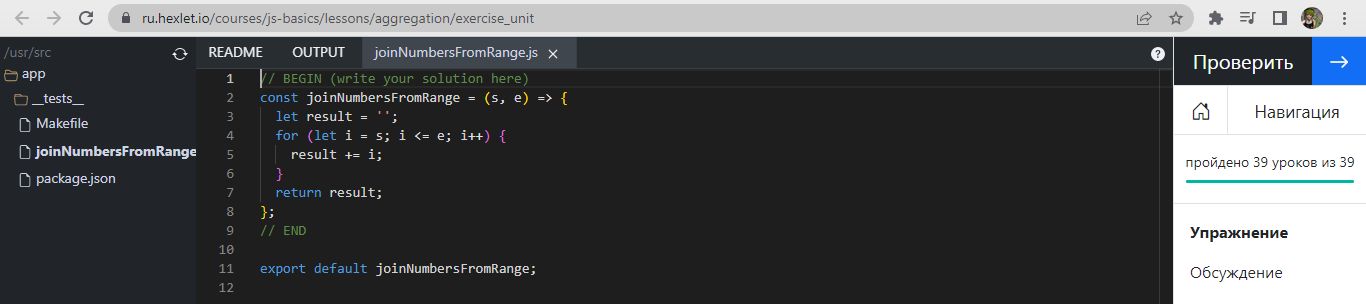
Решение:

34. Реализуйте функцию joinNumbersFromRange(), которая объединяет все числа из диапазона в строку и возвращает её:

joinNumbersFromRange(1, 1); // '1'

joinNumbersFromRange(2, 3); // '23'

joinNumbersFromRange(5, 10); // '5678910'

Решение:

35. Реализуйте функцию mySubstr(), которая извлекает из строки подстроку указанной длины. Она принимает на вход два аргумента (строку и длину) и возвращает подстроку, начиная с первого символа:

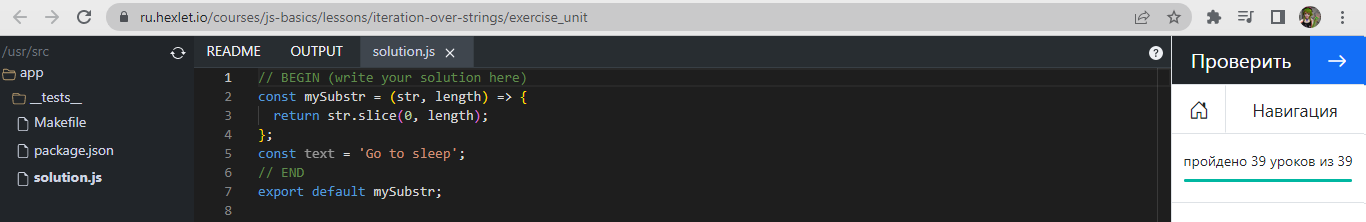
const text = 'If I look back I am lost';

console.log(mySubstr(text, 1)); # => 'I'

console.log(mySubstr(text, 7)); # => 'If I lo'

В этом задании аргумент, обозначающий длину извлекаемой подстроки, не может быть длиннее самой строки.

Используйте тот же подход, что в функции для переворота строки из урока: собирайте строку-результат в цикле, перебирая начальную строку до определённого момента.

Решение:

36. Функция countChars() из теории считает, сколько раз входит буква в предложение и при этом учитывает регистр букв. То есть A и a с её точки зрения разные символы. Реализуйте вариант этой же функции, так чтобы регистр букв был не важен:

countChars('HexlEt', 'e'); // 2

countChars('HexlEt', 'E'); // 2

Решение:

37. Напишите функцию makeItFunny(), которая принимает на вход строку и возвращает её копию, у которой каждый n-ный элемент переведен в верхний регистр. n – задается на входе в функцию.

Для определения каждого n-ного элемента понадобится остаток от деления %. Подумайте, как его можно использовать.

const text = 'I never look back';

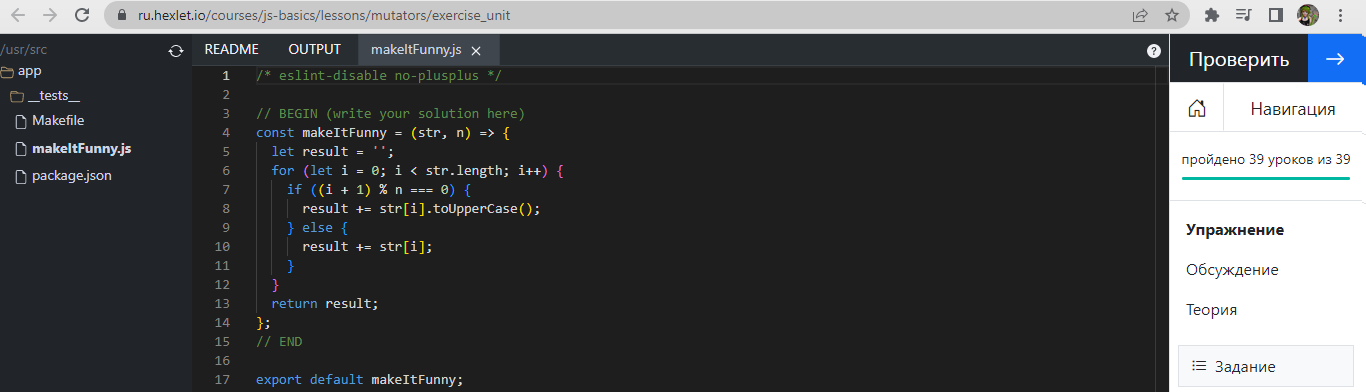
// Каждый третий элемент

makeItFunny(text, 3); // 'I NevEr LooK bAck'

Пример работы оператора остатка от деления:

console.log(20 % 3) // => 2

// Так как 3 \* 6 = 18

Решение:

38. Сэмвелл обнаружил, что его сообщения перехватываются и читаются в замке «Близнецы», поэтому его атаки перестали быть внезапными. Немного подумав, он разработал программу, которая будет шифровать передаваемые сообщения по следующему алгоритму: программа получает на вход строку и меняет местами в ней каждые 2 подряд идущих символа. Если длина строки нечётная, то последний символ остаётся на своём месте.

encrypt('move'); // 'omev'

encrypt('attack'); // 'taatkc'

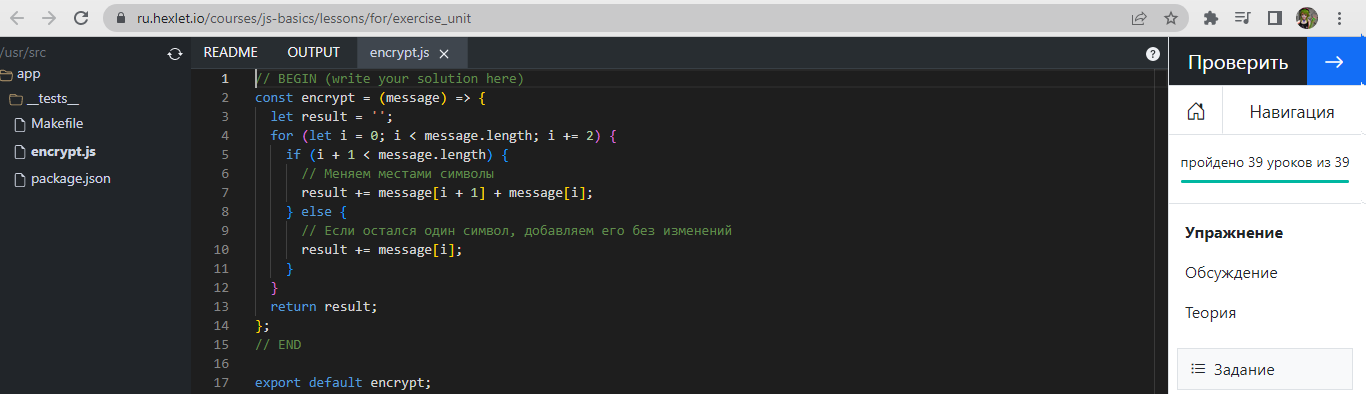
encrypt('car!'); // 'ac!r'

// Если длина строки нечётная,

// то последний символ остаётся на своём месте

encrypt('go!'); // 'og!'

Реализуйте функцию encrypt(), которая принимает на вход исходное сообщение и возвращает зашифрованное.

Решение:

39. В файле myMathModule.js:

Создайте функцию getTriangleArea(), которая принимает два аргумента h и b и вычисляет площадь треугольника по формуле A = 1/2 \* h \* b, где h — высота, а b — основание треугольника.

Вычисление площади треугольника

getTriangleArea(5, 10) === 25;

getTriangleArea(15, 12) === 90;

Экспортируйте функцию.

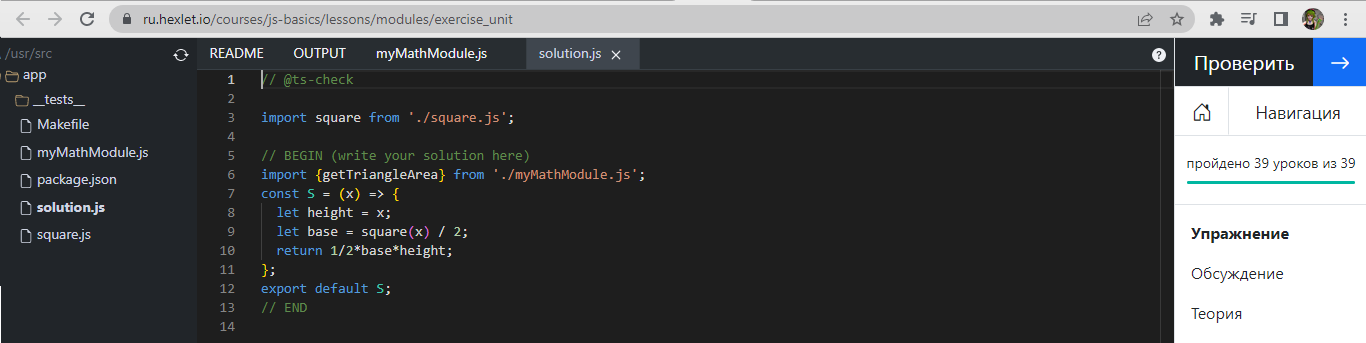
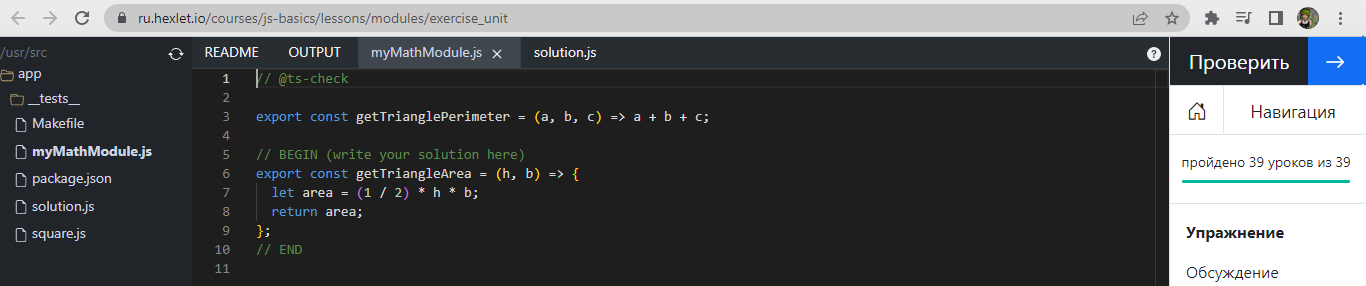
solution.js

В файле solution.js:

Импортируйте функцию getTriangleArea() из модуля myMathModule.

Создайте функцию, которая принимает аргумент n и возвращает площадь треугольника высотой n и основанием n2/2. Используйте функцию square() (принимает число и возвращает его квадрат).

Экспортируйте созданную функцию по умолчанию.

Решение: 

40. 40 урок не содержит упражнений.