WB School.
Курс «хардкорный JS. L1»: основы

Как делать задания

1. В заданиях никаких устных решений — только код. Одно решение — один файл с хорошо откомментированным кодом. Каждое решение или невозможность решения надо объяснить.-

2. Никаких сторонних библиотек и фреймворков, если только это специально не оговорено заданием. Только JavaScript.

3. Разрешается и приветствуется использование любых справочных ресурсов, привлечение сторонних экспертов и т.д. и т.п.

4. Основной критерий оценки — четкое понимание «как это работает». Некоторые задачи можно решить несколькими способами, в этом случае требуется привести максимально возможное количество вариантов.

5. Можно задавать вопросы, как по условию задач, так и об их решении. Идеальный вариант — продемонстрировать свои решения и получить максимальный фидбэк от опытных разработчиков Wildberries.

Задания

1. Задача о палиндроме: напишите функцию, которая проверяет, является ли заданная строка палиндромом. Палиндром — это строка, которая читается одинаково в обоих направлениях (например, «аргентина манит негра »).

2. Задача о странных числах: Напишите функцию, которая принимает число и возвращает true, если это число является странным, и false в противном случае. Странным числом считается число, которое равно сумме всех своих делителей, кроме самого себя.

3. Реализовать аналог библиотеки Math (можно назвать MathX) с базовым набором функций, используя замыкания:
   - вычисление N-го числа в ряду Фибоначчи
   - вычисление всех чисел в ряду Фибоначчи до числа N
   - вычисление N-вычисление всех простых чисел до числа N

   Будет плюсом, если задумаетесь и об оптимизации.

4. Разработать функцию, изменяющую окончание слов в зависимости от падежа. Например:

- 112 сообщения
- 12 сообщений
- 1 сообщение
- 1024 пользователя
- 1026 пользователей
- 121 пользователь`

Функцию надо упаковать в модуль.

5. Разработайте функцию преобразования JSON в связный список. На входе функция должна получать JSON, содержащий список объектов, на выходе объект, представляющий из себя односвязный список.

6. Задача о сортировке объектов: у вас есть массив объектов вида { name: 'John', age: 25 }. Напишите код, который сортирует этот массив по возрастанию возраста, а при равных возрастах сортирует по алфавиту по полю name.

7. Задача о коллекции функций: у вас есть массив функций, напишите код, который вызовет каждую функцию в этом массиве и выведет их порядковый номер. Однако, вызов каждой функции должен происходить только после вызова предыдущей функции.

Другими словами, нужно выполнить следующие шаги:

a. Вызвать первую функцию из массива.
b. После завершения работы первой функции вызвать вторую функцию.
c. После завершения работы второй функции вызвать третью функцию.
d. И так далее, пока все функции в массиве не будут вызваны по порядку.

8. Задача о замыканиях: напишите функцию, которая будет принимать массив функций и возвращать новую функцию, которая вызывает каждую функцию в этом массиве и возвращает массив результатов, полученных после вызова каждой функции.

9. Реализовать функцию конвертации JSON в строку (stringify ??)

10. Реализовать функцию конвертации строки в JSON со всеми необходимыми проверками и валидациями. (parse??)

11. Задача о замыканиях и области видимости: напишите функцию, которая возвращает другую функцию. Внутренняя функция должна иметь доступ к переменной, определенной во внешней функции, даже после того, как внешняя функция завершила свое выполнение.

12. Задача на работу с объектами: создайте объект, представляющий собой книгу. Объект должен иметь свойства, такие как: название книги, автор и год издания. Напишите методы для получения и изменения значений свойств книги.

13. Задача на классы и наследование: создайте базовый класс Shape (фигура), который имеет методы для расчета площади и периметра. Затем создайте подклассы,

представляющие различные фигуры, такие как прямоугольник, круг и треугольник. Реализуйте методы расчета площади и периметра для каждой фигуры.

14. Задача на промисы: напишите функцию, которая принимает URL изображения и возвращает промис, который разрешается с данными об изображении, когда оно загружено. Когда говорится "промис разрешается с данными об изображении", это означает, что промис должен быть успешно выполнен (resolved) с данными об изображении после того, как изображение будет загружено.

15. Задача на асинхронность: напишите асинхронную функцию, которая использует ключевое слово await для ожидания выполнения других асинхронных операций, и возвращает результат выполнения.

16. Задача на модули и использование внешних библиотек: напишите модуль, который экспортирует функцию для работы с датами. Внутри модуля используйте внешнюю библиотеку Moment.js для удобной работы с датами.

17. Необходимо реализовать простое поле ввода адреса с функцией геокодинга: пользователь вводит данные в поле с помощью одного из геоинформационных сервисов (Яндекс.Карты, ДаДата, GraphHopper), подбирается адрес. Найденные данные должны отображаться в выпадающем списке, из которого можно выбрать подходящее значение.

Реализовать дебоунсинг и защиту от троттлинга с помощью замыканий.

18. Подсчитать максимальный объем данных, который можно записать в localStorage вашего браузера.

19. Реализовать виджет, отображающий список постов из любого паблика в VK (подойдет любой паблик, где постов очень много). Например, с помощью этой функции API VK. Виджет должен иметь фиксированные размеры и возможность прокрутки. При прокрутке содержимого виджета до конца должны подгружаться новые посты. Необходимо реализовать возможность кэширования уже загруженных данных: если пользователь закрыл страницу, а потом снова открыл ее, виджет должен отображать все загруженные ранее данные (новые данные должны подгружаться из учетом уже загруженных ранее).

    При переполнении localStorage, данные, загруженные последними должны вытеснять данные загруженные первыми.

20. Реализовать функцию подсчета объема памяти занимаемого данными в LocalStorage для предыдущей задачи. При изменении данных в localStorage в консоль должен выводиться объем занятой памяти / максимальный размер хранилища.

21. Вычислить размер коллстэка в основных браузерах: Chrome, Firefox, Opera и Safari (если есть возможность).

22. Посчитайте сколько раз можно вызвать функцию document.write() внутри document.write(). Объясните результат.

23. Анализатор сложности пароля: создайте функцию, которая оценивает сложность введенного пользователем пароля. Необходимо анализировать длину пароля, использование различных символов, наличие чисел и букв в разных регистрах. Выведите пользователю оценку сложности пароля и предложите улучшения, если пароль слишком слабый.

24. Разработайте страницу, отображающую таблицу с данными. Данные необходимо подгружать из этого источника.

Требования:

- данные должны загружаться при загрузке страницы
- необходимо реализовать сортировку по убыванию и по возрастания для всех колонок
- необходимо реализовать клиентскую пагинацию (50 элементов на странице)

25. Задача: Создать и добавить стиль для элемента: Напишите функцию, которая создает новый элемент, добавляет его в DOM и устанавливает для него стиль с помощью CSS.
26. Задача: Рекурсивный обход дерева DOM:: Напишите функцию, которая рекурсивно обходит дерево DOM, начиная с указанного элемента, и выполняет определенное действие с каждым узлом (например, выводить информацию о теге в консоль).
27. Задача: Добавить анимацию для элемента: Напишите функцию, которая добавляет анимацию для элемента на веб-странице, например, плавное изменение его положения или размера.
28. Задача: Создать и добавить элемент с использованием шаблонов: Напишите функцию, которая создает новый элемент с использованием шаблонов (например, с помощью тега <template>) и добавляет его в DOM.
29. Задача: Взаимодействие с формами: Напишите функцию, которая получает данные из формы на веб-странице и выполняет определенные действия с этими данными, например, отправляет их на сервер или отображает всплывающее окно с результатами.