Цель урока:

Вы научитесь выполнять базовые действия с массивами: удаление, добавление, изменение элементов. Также мы разберем такие темы, как перебор массива, сортировка массива разными способами, объединение нескольких массивов в один.

Содержание урока:

1. Что такое массивы и как их создавать?
2. Добавление, удаление и изменение элементов массива
3. Перебор массивов с помощью for, for of и forEach
4. Работа с методами массивов: map, filter, find, findIndex, some, every
5. Метод массивов reduce
6. Метод массивов sort
7. Методы массивов: splice, slice, indexOf, includes, join, reverse
8. Объединение нескольких массивов в один

Дополнительные материалы:

1. Spread-оператор в JavaScript (...): https://habr.com/ru/post/34871...
2. Массивы и их методы: https://developer.mozilla.org/...
3. Как работает reduce в javascript: https://medium.com/@stasonmars...
4. Метод sort в JavaScript: https://developer.mozilla.org/...
5. Методы массивов (forEach, map, filter, reduce, find, findIndex): https://youtu.be/nEabP9CYCAQ
6. Всё о деструктуризации: https://youtu.be/wWYokY0Pt2M

UPD: метод splice вторым параметром принимает количество удаляемых элементов! Описание метода

Задание #1

Представьте, что мы разрабатываем систему очередей для какого-либо заведения. Например, для почты. Для этого мы создали массив peopleWaiting, который отображает текущий статус очереди.

```
const peopleWaiting = ['Кристина', 'Олег', 'Кирилл', 'Мария', 'Светлана', 'Артем', 'Глеб'];
```

По логике, следующую посылку по почте получит Кристина, после нее уже 1-м в очереди будет Олег.

Сейчас вам необходимо реализовать следующую логику в коде шаг за шагом:

1. Кристина и Олег получили посылки и ушли из очереди. Вам необходимо удалить их из массива.
2. Теперь подошла очередь к Кириллу. И неожиданно сотрудница почты говорит, что скоро у них обеденный перерыв и она успеет обслужить только Кирилла. Поэтому все остальные люди, стоящие за Кириллом, решили не ждать, когда закончится обед и просто ушли из отделения почты. Вам необходимо сначала удалить Кирилла из массива peopleWaiting, а затем удалить людей, которые не успели получить посылки.

Когда какой-либо человек получает посылку, необходимо вывести в модальном окне сообщение “name получил(а) посылку. В очереди осталось length человек.” (Замените name на имя человека, получившего посылку, а length - на количество человек, которые остались в очереди).

Если же человек не получил посылку и ушел из очереди, то выведите в модальном окне через alert сообщение “name не получил(а) посылку и ушел(ла) из очереди”.

Рекомендуется создать 2 функции: **giveParcel** - для выдачи посылки и удаления клиента из начала массива, **leaveQueueWithoutParcel** - для удаления клиента, который не получил посылку из конца списка.

Задание #2

Вам необходимо создать функцию **getSumOfSequence**, которая будет создавать массив из чисел и высчитывать сумму первого и последнего элементов массива.

Функция принимает в себя один параметр **number**. Используя этот параметр, нужно создать массив из чисел от 1 до **number**. Для заполнения массива элементами используйте цикл for. (Например, если был передан number равный 5, то массив должен выглядеть следующим образом: [1, 2, 3, 4, 5])

Функция **getSumOfSequence** должна возвращать сумму первого и последнего элементов итогового массива.

Пример:

```
getSumOfSequence(5) = 6 // [1, 2, 3, 4, 5], 1+5=6
```

Задание #3

Представьте, что у вас есть друг Алексей, который является владельцем кофейни. Он хочет сделать сайт для этой кофейни, чтобы люди могли заказывать доставку кофе на дом. Самая главная функция, которую хочет видеть Алексей, это поиск кофе по названию. Ваш друг просит вас помочь ему с разработкой сайта.

Изначально у вас есть массив coffees, в котором хранятся все виды кофе, которые есть в кофейне у Алексея. Напитки распределены в порядке популярности среди клиентов. Т.е. кофе “Latte” - самый популярный.

```
const coffees = ['Latte', 'Cappuccino', 'Americano'];
```

С помощью функции **prompt** вам необходимо запрашивать название кофе. Передайте в prompt сообщение “Поиск кофе по названию:”. Сохраните значение, которое ввел пользователь в переменную coffeeName.

Если введенное пользователем название кофе существует, то отобразите сообщение с помощью **alert** “Держите ваш любимый кофе coffee. Он number-й по популярности в нашей кофейне.”, где coffee - название найденного кофе, number - номер найденного кофе в массиве coffees. Если же кофе не был найден, то отобразите сообщение “К сожалению, такого вида кофе нет в наличии”.

Условия:

1. Название кофе должно быть регистронезависимым. Т.е. если пользователь напечатает “lATte”, то должен показаться результат с кофе “Latte”
2. Необходимо использовать findIndex.

Задание #4

Вы разработали главную функциональность, которую хотел видеть Алексей на своем сайте. Он говорит вам большое спасибо, потому что благодаря сайту в кофейне увеличилось количество продаж. Теперь Алексей захотел поднять цену на кофе, чтобы прибыли было еще больше.

У вас изначально есть 2 массива: coffees (хранит названия кофе) и prices (хранит цены на кофе). Индексы для названий кофе и цен взаимосвязаны. Т.е. “Latte” стоит 1.5 евро, “Cappuccino” - 1 евро, и т.д.

```
const coffees = ['Latte', 'Cappuccino', 'Americano'];
const prices = [1.5, 1, 2];
```

Вам **необходимо создать новый массив updatedPrices**, который будет содержать те же числа из массива prices, но только увеличенные на 0.5.

После подъема цен вам необходимо уведомить об этом клиентов кофейни, поэтому выведите для каждого типа кофе сообщение через alert “Кофе name сейчас стоит price евро”, где name - название кофе, а price - обновленная цена на кофе.

Условия:

1. Необходимо использовать методы массивов map и forEach

Задание #5

Последнее, что хочет видеть Алексей на своем сайте - это система оценок работы кофейни. Владельцу необходимо знать мнение клиентов, чтобы исправлять недочеты в своей работе.

Для начала вам необходимо **создать пустой массив clientsEstimations**, в котором будут хранится оценки клиентов кофейни. Также **создайте функцию askClientToGiveEstimation**, которая должна выводить сообщение “Как вы оцениваете нашу кофейню от 1 до 10?” через prompt. Конечный результат, который введет пользователь, должен быть типом данных number. Если было введено число от 1 до 10, то добавьте эту оценку в массив clientsEstimations, иначе же никаких действий не совершайте.

Для добавления оценок вызовите функцию askClientToGiveEstimation 5 раз. Рекомендуется это сделать через цикл for.

После того, как оценки будут добавлены, вам необходимо посчитать положительные и отрицательных оценки. Положительной оценкой является число больше 5, отрицательной - число, меньшее за 5 либо равное 5. **Выведите через alert** сообщение “Всего положительных оценок: goodEstimations; Всего отрицательных оценок: notGoodEstimations”, где goodEstimations - количество положительных оценок, а notGoodEstimations - количество отрицательных оценок.

Условия:

1. В решение должен быть использован метод массивов filter.

Задание #6

Вам дан массив чисел.

```
const numbers = [10, 4, 100, -5, 54, 2];
```

Необходимо посчитать сумму кубов всех чисел в массиве. Например, для массива [1, 5, 9] должен быть следующий результат:

1^3 + 5^3 + 9^3 = 1 + 125 + 729 = **855**

Решите данную задачу 4-мя способами:

1. Через цикл for
2. Через цикл for of
3. Через метод forEach
4. Через метод reduce

Для данного массива numbers сумма кубов должна равняться 1158411.

Задание #7 (*дополнительное*)

Представьте, что ваш друг - владелец любительского футбольного клуба. Игроки команды забивают много голов и сотрудникам клуба надоело каждый раз после матча считать всю статистику. Поэтому они решили попросить вас как веб-разработчика помочь решить эту проблему. Вам требуется оптимизировать задачу подсчета статистики с помощью языка программирования JavaScript.

У вас есть массив goals, в котором каждый индекс означает сыгранный матч, а число под индексом - количество голов забитых в конкретном матче.

```
const goals = [8, 1, 1, 3, 2, -1, 5];
```

То есть в 1-м матче команда забила 8 мячей, во 2-м - один и т.д. Если в массиве есть отрицательное число, то значит, что команде было засчитано автоматическое поражение.

Используя данный массив goals вам необходимо вывести следующую статистику за весь сезон.

1. **Самый результативный матч по количеству голов**. Выведите в модальном окне через alert сообщение “Самый результативный матч был под номером number. В нем было забито numberOfGoals гол(ов).” (замените number на порядковый номер матча, а numberOfGoals - на количество голов в самом результативном матче). Если самых результативных матчей несколько, то выведите первый из них.
2. **Самые нерезультативные игры**. В массиве нужно найти **все** самые нерезультативные игры (все матчи, у которых количество голов между собой совпадает и является минимальным). В данном массиве goals это будут 2 матча под номерами 2 и 3 с количеством голов по 1. Выведите через alert сообщение “Самые нерезультативные матчи были под номерами numbers. В каждом из них было забито по numberOfGoals мячу(а).” (замените numbers на порядковые номера матчей и отобразите их через запятую, а numberOfGoals - на количество голов в самом нерезультативном матче). Не берите в учет игры с автоматическим поражением.
3. **Общее количество голов за сезон**. Не берите в учет игры с автоматическим поражением. Выведите сообщение через alert “Общее количество голов за сезон равно numberOfGoals” (замените numberOfGoals на число общее количества голов за сезон).
4. **Были ли автоматические поражения**. Если были, то выведите сообщение через alert “Были автоматические поражения: да”, иначе “Были автоматические поражения: нет”.
5. **Среднее количество голов за матч**. Выведите сообщение через alert “Среднее количество голов за матч равно numberOfGoals” (замените numberOfGoals на среднее количество голов за матч).
6. **Отсортируйте голы в порядке возрастания** и выведите все результаты через запятую в модальном окне alert. Массив goals не должен быть изменен.

Задание #8 (*дополнительное*)

Ваша задача состоит в том, чтобы **написать функцию**, которая будет работать как мини-калькулятор математических выражений. Назовите данную функцию **getMathResult**. getMathResult принимает в себя 1 параметр expression, который является массивом и всегда состоит из 3-х элементов.

Первый и 3-й элементы в массиве expression - это числа, которые могут записывать как тип данных number либо string (например, 100 или ‘100’). 2-й элемент - это математический знак, который является типом данных string. Математический знак может быть исключительно следующими строками: “>”, “<”, “=”, “+”, “-”, “*”, “/”. Если был знак,

которого не существует в данной последовательности, то функция getMathResult должна возвращать ошибку в виде строки “Ошибка”.

Примеры результатов функции getMathResult:

```
getMathResult(['200', '+', 300]); // 500
getMathResult(['20', '-', '5']); // 15
getMathResult([100, '/', 100]); // 1
getMathResult([2, '-', 2]); // 0
getMathResult(['5', '>', '10']); // false
getMathResult(['5', '<', '10']); // true
getMathResult(['1', '=', 1]); // true
getMathResult(['1', '**', 1]); // 'Ошибка'
```

Также учтите, что в массив expression иногда может быть передано больше 3-х параметров, и это конечно же неверно. Но если все же такое было сделано, вам необходимо удалить лишние элементы в массиве, которые нельзя преобразовать к числу, которые являются ложными. Если же длина массива expression меньше 3, то функция getMathResult должна возвращать ошибку в виде строки “Ошибка”.

Например, если expression был передан как данный массив:

```
['100', 'hello', 'javascript', , 'help200', '+', 4]
```

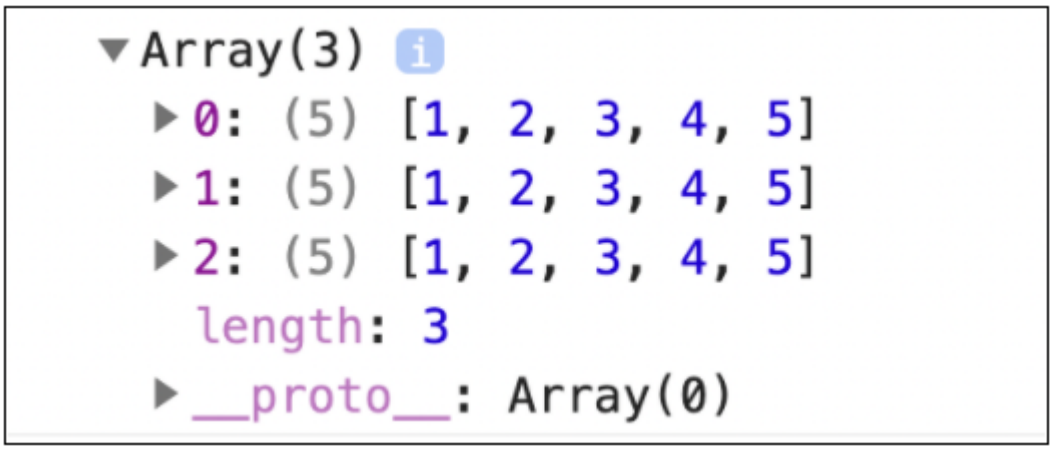
то в итоге он должен быть преобразован к:

```
['100', '+', 4]
```

Примечание. В решении нельзя использовать eval

Задание #9 (*дополнительное*)

С помощью циклов for **вам необходимо создать матрицу 3x5**. В итоге она должна выглядеть следующим образом:



Подсказка: как видите, в итоге получился массив из 3-х элементов, внутри которого хранятся массивы, каждый из которых имеет по 5 элементов. Это можно реализовать с помощью 2-х циклов for. Внутрь одного цикла for необходимо поместить еще один цикл.

Задание #10 (*дополнительное*)

Вам дан массив, элементами которого являются массивы.

```
const matrix = [
       [ 1, 2, 3 ],
       [ 4, 5, 6 ],
       [ 7, 8, 9 ],
];
```

Вам **необходимо сделать** из элементов вложенных массивов **один массив**. Итоговый результат должен выглядеть следующим образом: [1, 2, 3, 4, 5, 6, 7, 8, 9].

Условия:

1. Нельзя использовать метод массивов flat.
2. Необходимо, чтобы в решении использовался метод concat либо spread-оператор