1. Чем отличаются куки, localStorage и sessionStorage? Что стоит использовать, если нам нужно надолго сохранить много пользовательских данных?  
   Чтобы долго хранить пользовательские данные нужно использовать Local Storage.  
     
   Есть три способа хранения данных на клиенте:

Local Storage (локальное хранилище)

- Хранит данные бессрочно.

- Очищается только с помощью JavaScript или очистки кэша браузера.

- Хранит данные объёмом до 5 МБ, это самый большой объём из трёх вариантов хранилища.

- Не поддерживается старыми браузерами, например, IE 7 и ниже.

Session Storage (сессионное хранилище)

- Хранит данные, пока продолжается текущая сессия. Когда пользователь закрывает браузер, данные становятся недоступными.

- Используется контекст браузера верхнего уровня, поэтому каждая вкладка браузера хранит уникальные данные.

- Объём данных больше чем в Cookie.

- Не поддерживается старыми браузерами, например, IE 7 и ниже.

Cookie (куки)

- Хранит данные, которые можно передавать на сервер через заголовки.

- Локальное и сессионное хранилище доступны только на клиентской стороне, а cookie можно передавать на сервер.

- Срок хранения устанавливается при создании cookie.

- Объём данных не превышает 4 Кбайт.

1. Как добавить и получить значение из веб-хранилища?

`setItem()` — добавляет пару ключ-значение в локальное веб-хранилище

`getItem()` — получает значение по ключу

1. Придумайте еще минимум 3 ситуации помимо предложенных в уроке, для чего может быть нужно сохранять данные пользователя и какие?

Пример из урока: запомнить, что пользователь уже залогинился, что у него лежит в корзине или в каком разделе сайта он сейчас находится  
  
а) для того, чтобы в игре сохранить его текущий уровень  
б) для сохранения пользовательских настроек  
в) на сайте-переводчике сохранение слов, которые человек хочет выучить

1. Как сделать валидацию номера кредитной карты?  
     
   Через регулярные выражения:

    <script>

        function ValidateCard(cardField) {

            var cardFormat =

                /^(?:4[0-9]{12}(?:[0-9]{3})?|5[1-5][0-9]{14}|6(?:011|5[0-9][0-9])[0-9]{12}|3[47][0-9]{13}|3(?:0[0-5]|[68][0-9])[0-9]{11}|(?:2131|1800|35\d{3})\d{11})$/;

            if (cardField.match(cardFormat)) {

                console.log("ok");  
 return true;

            } else {

                alert("Ваш номер карты введен неверно!");

                return false;

            }

        }

        ValidateCard ('4000001934567899')

    </script>

Через метод test:

    <script>

        function ValidateCard(cardField) {

            var cardFormat =

                /^(?:4[0-9]{12}(?:[0-9]{3})?|5[1-5][0-9]{14}|6(?:011|5[0-9][0-9])[0-9]{12}|3[47][0-9]{13}|3(?:0[0-5]|[68][0-9])[0-9]{11}|(?:2131|1800|35\d{3})\d{11})$/;

            if (cardFormat.test(cardField)) {

                console.log("ok");

                return true;

            } else {

                alert("Ваш номер карты введен неверно!");

                return false;

            }

        }

        ValidateCard('4000001934567899')

    </script>

1. Как сделать input, который будет принимать только числа (минимум 2 способа)?  
     
   а) Через type  
   <input type=’number’>  
   б) Через регулярное выражение

    <input type="text" id="input">

    <script>

        let input = document.querySelector('#input');

        input.oninput = function () {

            input.value = input.value.replace(/[^\d]/g, '')

        };

    </script>

1. Найдите регулярное выражение для ФИО на русском языке  
     
   /^([А-ЯA-Z]|[А-ЯA-Z][\x27а-яa-z]{1,}|[А-ЯA-Z][\x27а-яa-z]{1,}\-([А-ЯA-Z][\x27а-яa-z]{1,}|(оглы)|(кызы)))\040[А-ЯA-Z][\x27а-яa-z]{1,}(\040[А-ЯA-Z][\x27а-яa-z]{1,})?$/
2. Как должно выглядеть невалидное поле, чтобы пользователю было понятно, что в него вводить? Как показать пользователю, что не так, если ввод некорректен (теги и атрибуты html, псевдоклассы css) ? вопрос теоретический, код писать не надо  
     
   В CSS существует четыре специальных псевдокласса, применимых к полям формы: :valid (валидное поле), :invalid (невалидное), :required (обязательное) и :optional (необязательное). Их можно использовать, чтобы добавлять некоторые — хотя и весьма ограниченные — подсказки пользователям, заполняющим форму.

Используя :valid и :invalid, мы можем показать пользователю, правильно ли заполнено поле по мере ввода.  
input:required:valid { border-color: green;}

1. Какие есть недостатки у стандартного способа задания валидации через HTML5?

- Ошибки выдаются по одной

- Свойства ограничены, примитивность