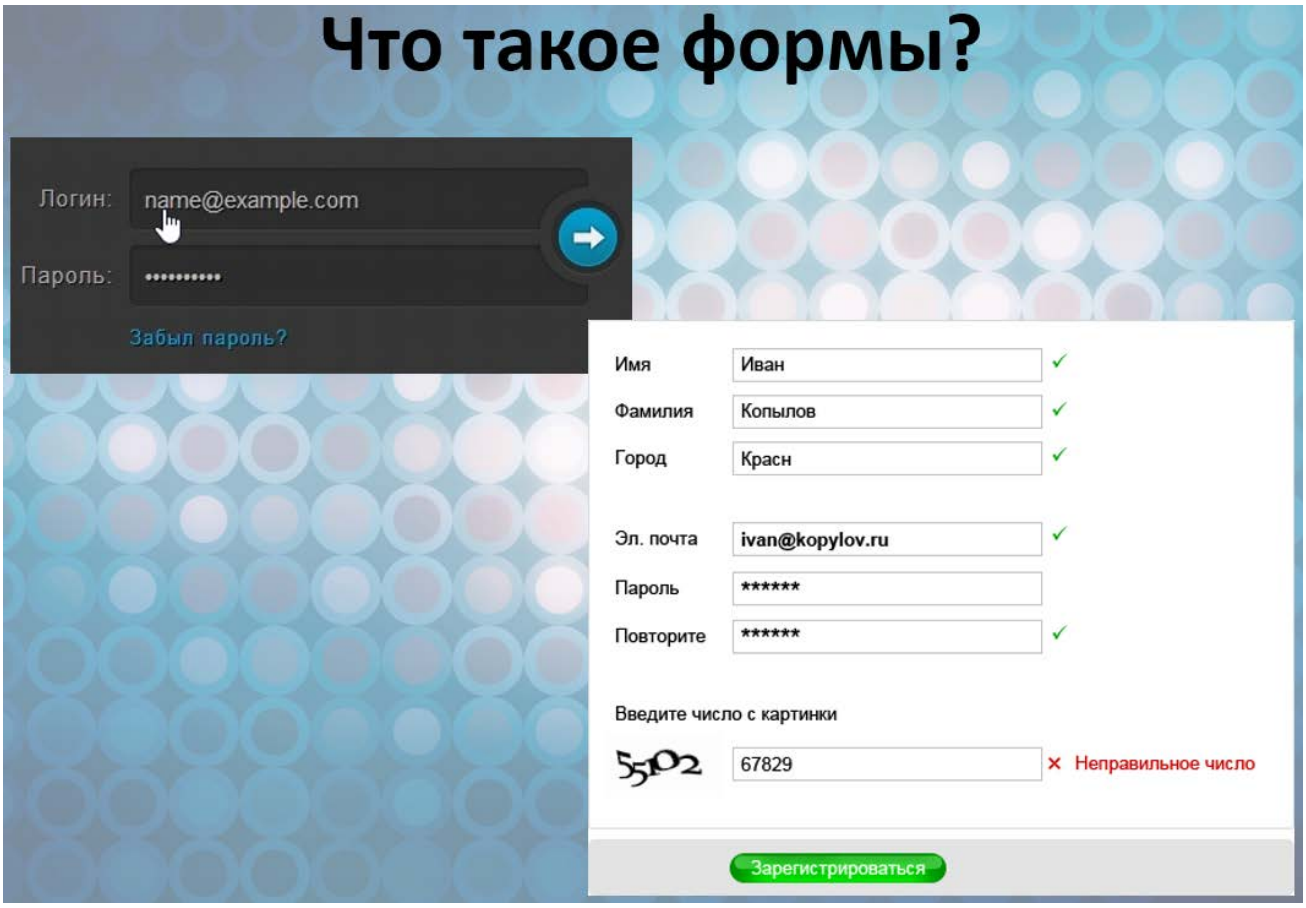
Web-программирование

Практическое занятие 6

Примечание: в данном занятии рассматриваются формы, которые, как правило, работают в паре с серверной частью, запоминающей введённые данные. Но, так как этой темы мы будем касаться позже, цель этого занятия – познакомится с основными инструментами формы и научится их размещать. Содержимое формы будет использовано как вставки в придуманную учащимся историю.

1. Сегодня мы с вами поговорим о формах
2. Что же такое форма? Форма – это элемент страницы, служащий для заполнения какими-либо данными пользователем. Такие формы мы встречаем на сайтах, когда регистрируемся и логинимся. Как правило, введённая информация отправляется на сервер и там обрабатывается.



3. Тег <form> устанавливает форму на веб-странице. Форма предназначена для обмена данными между пользователем и сервером. Область применения форм не ограничена отправкой данных на сервер, с помощью клиентских скриптов можно получить доступ к любому элементу формы, изменять его и применять по своему усмотрению.

```
<body>
    <form action="" onsubmit="submitAction()" name="nameForm">
        <input type="submit" name="but2"/>
    </form>
    <script type="text/javascript" src="js1.js"></script>
</body>
```

Документ может содержать любое количество форм, но одновременно на сервер может быть отправлена только одна форма. По этой причине данные форм должны быть независимы друг от друга.

Для отправки формы на сервер используется кнопка Submit, того же можно добиться, если нажать клавишу Enter в пределах формы. Если кнопка Submit отсутствует в форме, клавиша Enter имитирует ее использование.

Так как в этот раз по-настоящему форму мы никуда отправлять не будем, поле action мы оставим пустым. Вместо этого на событие onsubmit мы будем выполнять свой скрипт.

Событие onsubmit возникает при отправке формы, это обычно происходит, когда пользователь нажимает специальную кнопку Submit.

Чтобы поместить элемент на эту форму, нужно воспользоваться тегом <input>

Тег <input> является одним из разносторонних элементов формы и позволяет создавать разные элементы интерфейса и обеспечить взаимодействие с пользователем. Главным образом <input> предназначен для создания текстовых полей, различных кнопок, переключателей и флажков. Хотя элемент <input> не требуется помещать внутрь контейнера <form>, определяющего форму, но если введенные пользователем данные должны быть отправлены на сервер, где их обрабатывает серверная программа, то указывать <form> обязательно. То же самое обстоит и в случае обработки данных с помощью клиентских приложений, например, скриптов на языке JavaScript.

Повторите код на слайде и создайте в JS-файле функцию, выводящую алерт по нажатию кнопки submit.

```
function submitAction(){
    alert('Отправлено!');
}
```

4. Основной атрибут тега <input>, определяющий вид элемента — type. Он позволяет задавать следующие элементы формы: текстовое поле (text), поле с паролем (password), переключатель (radio), флажок (checkbox), скрытое поле (hidden), кнопка (button), кнопка для отправки формы (submit), кнопка для очистки формы (reset), поле для отправки файла (file) и кнопка с изображением (image). Для каждого элемента существует свой список атрибутов, которые определяют его вид и характеристики. Кроме того, в HTML5 добавлено еще более десятка новых элементов.
Запишите эти ключевые слова, позволяющие создавать различные элементы.

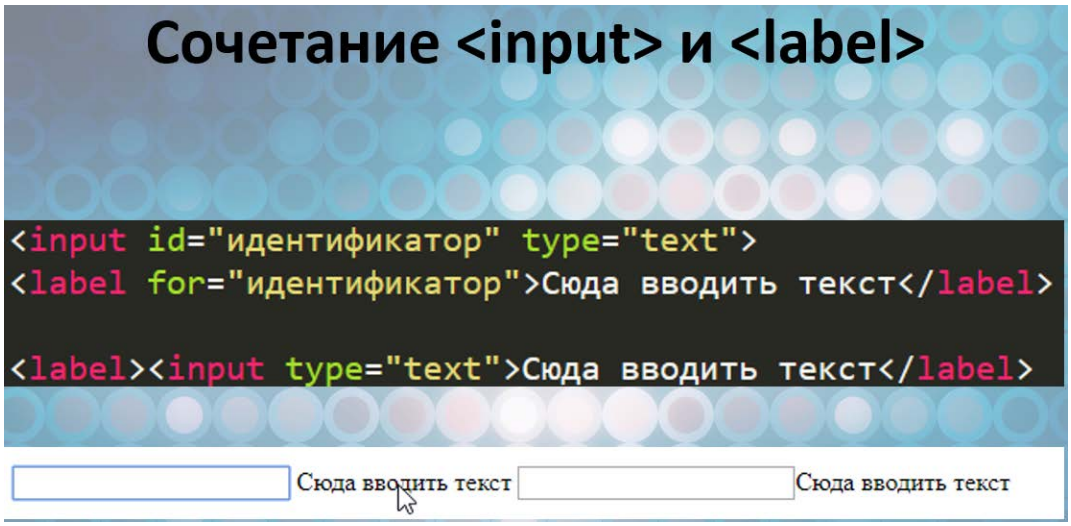
Разновидности type у <input>

Тип	Описание	Вид
button	Кнопка.	Кнопка
checkbox	Флажки. Позволяют выбрать более одного варианта из предложенных.	☑ Пиво ☐ Чай ☐ Кофе
file	Поле для ввода имени файла, который пересылается на сервер.	Выберите файл Файл не выбран
hidden	Скрытое поле. Оно никак не отображается на веб-странице.	
image	Поле с изображением. При нажатии на рисунок данные формы отправляются на сервер.	Отправить
password	Обычное текстовое поле, но отличается от него тем, что все символы показываются звездочками. Предназначено для того, чтобы никто не подглядел вводимый пароль.	••••••••••
radio	Переключатели. Используются, когда следует выбрать один вариант из нескольких предложенных.	○ Пиво ○ Чай ◉ Кофе
reset	Кнопка для возвращения данных формы в первоначальное значение.	Сбросить
submit	Кнопка для отправки данных формы на сервер.	Отправить
text	Текстовое поле. Предназначено для ввода символов с помощью клавиатуры.	ФЫВАПРОЛДЖ

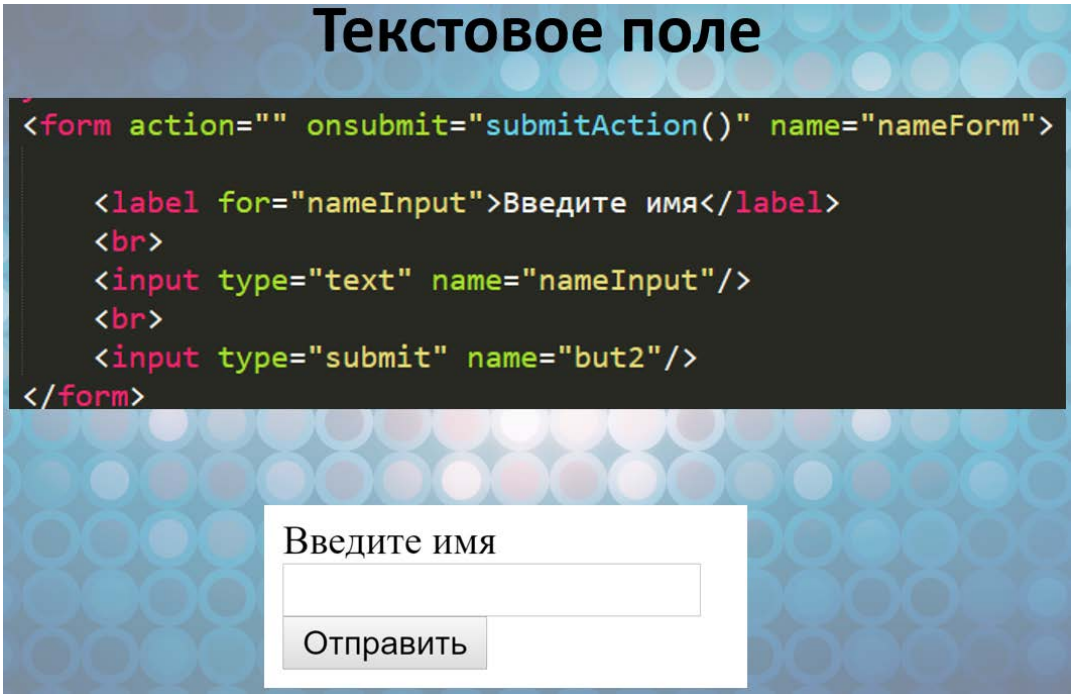
5. Тег <label> устанавливает связь между определенной меткой, в качестве которой обычно выступает текст, и элементом формы (<input>, <select>, <textarea>). Такая связь необходима, чтобы изменять значения элементов формы при нажатии курсором мыши на текст. Кроме того, с помощью <label> можно устанавливать горячие клавиши на клавиатуре и переходить на активный элемент подобно ссылкам.

   Существует два способа связывания объекта и метки. Первый заключается в использовании идентификатора id внутри элемента формы и указании его имени в качестве атрибута for тега <label>. При втором способе элемент формы помещается внутрь контейнера <label>.

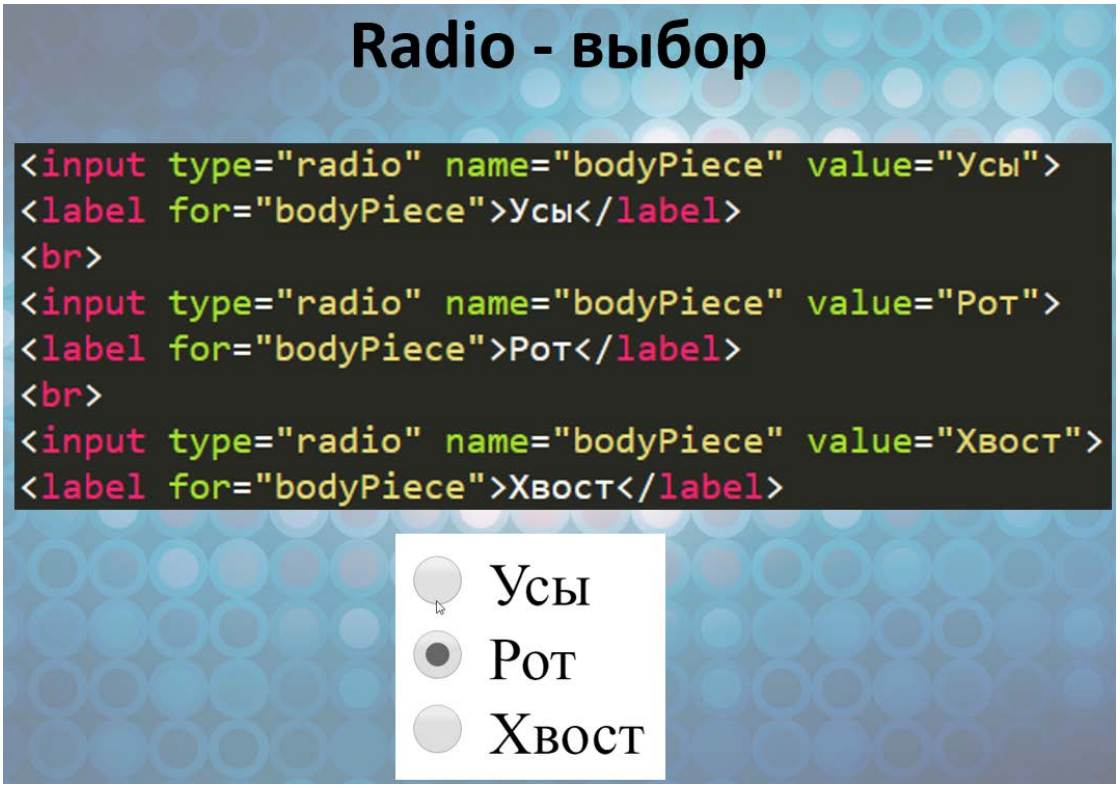
   Попробуйте оба варианта, добавив код на страницу. Нажав на текст, можно заметить, что соответствующее поле ввода получило фокус.



6. Создайте надпись «Введите имя» и текстовое поле и привяжите их друг к другу.



7. Радио-кнопки – это несколько объединённых в группу вариантов, из которых выбрать можно только один. Написаны эти варианты могут быть в любом месте документа, но объединяет их общий параметр name. Параметр value позволит нам позже понять, какой же из них был выбран.



8. Повторите слайд, не забывайте назначать новые имена новым элементам, если копируете текст.

Введите имя

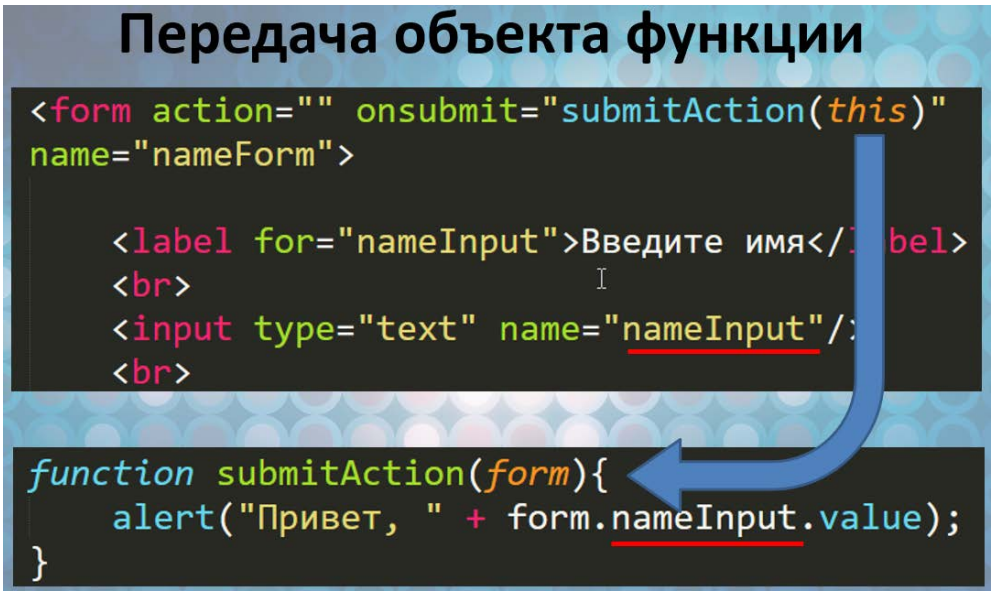
Усы
Рот
Хвост

Введите число

Сосисочки
Котлетки
Пюрешка

Отправить

9. Любой элемент страницы мы можем найти командой getElementById, но есть инструмент лучше – передача элемента как параметра. Если сработавшее событие относится к этому элементу, тогда мы можем использовать ключевое слово this, чтобы передать тот элемент. Тогда в первый параметр нашей функции запишется переданный элемент.
   Благодаря этому, мы можем обращаться к элементам внутри формы, указывая значение параметра name через точку.
   Повторите код. В форме введите имя и нажмите «отправить». Должно вывестись сообщение с текстом, введённым в эту форму.



10. Сама форма обычно предназначена для получения от пользователя информации для дальнейшей пересылки её на сервер, где данные формы принимает программа-обработчик. Такая программа может быть написана на любом серверном языке программирования вроде PHP, Perl и др. Адрес программы указывается в атрибуте action тега <form>. В этом примере данные формы, обозначенные атрибутом name (login и password), будут переданы в файл по адресу /example/handler.php. Если атрибут action не указывать, то передача происходит на адрес текущей страницы.

    Передача на сервер происходит двумя разными методами: GET и POST, для задания метода в теге <form> используется атрибут method, а его значениями выступают ключевые слова get и post. Если атрибут method не задан, то по умолчанию данные отправляются на сервер методом GET.

    Какой метод используется легко определить по адресной строке браузера. Если в ней появился вопросительный знак, то это точно GET.

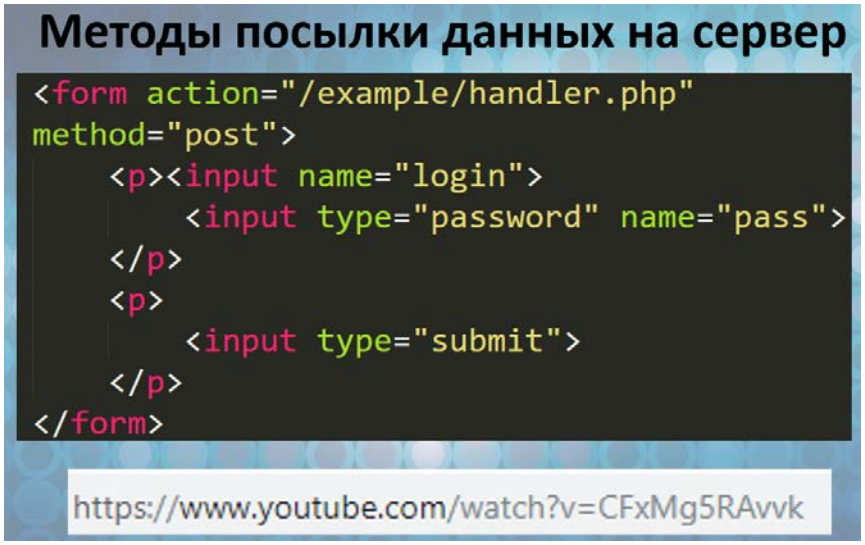
    GET

    Передача небольших текстовых данных на сервер; поиск по сайту.

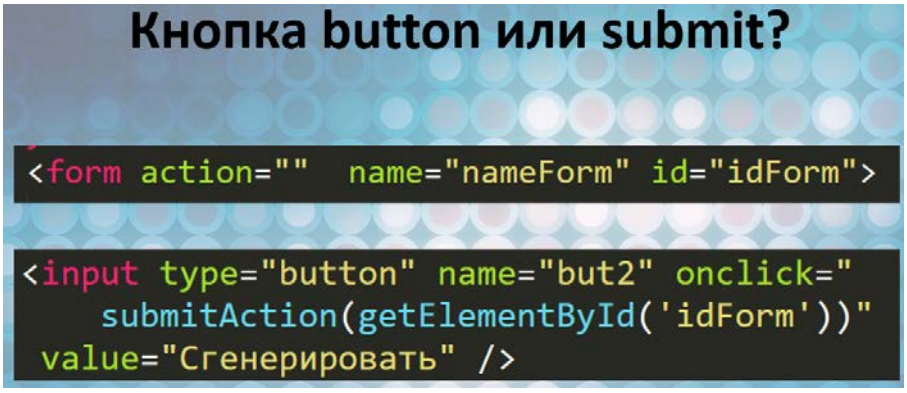
    Поисковые системы, формы поиска по сайту всегда отправляются методом GET, это позволяет делиться результатами поиска с друзьями, слать ссылку по почте или выкладывать её на форуме.

    POST

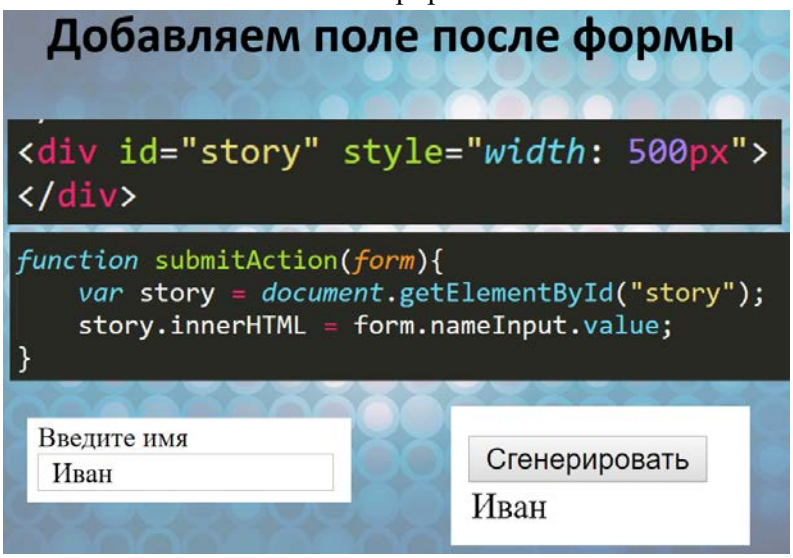
    Пересылка файлов (фотографий, архивов, программ и др.); отправка комментариев; добавление и редактирование сообщений на форуме, блоге.



11. Элемент input с типом submit посылает форму по указанному адресу и перезагружает страницу. Нам перезагрузка страницы не нужна, поэтому мы вместо кнопки submit сделаем простую кнопку типа button. По клику будем запускать всё ту же функцию, но, чтобы передать ей элемент «форма» как параметр, воспользуемся командой getElementById, с помощью которой найдём форму по id. Не забудьте указать форме этот id, параметр value указывает надпись на кнопке.



12. С помощью полученных сегодня знаний, мы напишем генератор историй. После формы вставьте элемент div, в который мы будем записывать историю. Повторите код, проверьте, что на экране после нажатия кнопки «Сгенерировать» появляется значение первого поля.



13. Самостоятельная работа заключается в следующем. Нужно придумать историю, в которой будут введённые в форму значения. Придумайте необычную историю, в которой эти значения используются интересным способом. После того, как закончите, дайте попробовать сгенерировать историю другим учащимся в группе.
    Задача со звёздочкой – сделать так, чтобы не выводились большие буквы там, где их быть не должно.