

# Лабораторная работа №1 по JavaScript

Добавление javascript в html

**Для добавления сценария на страницу HTML используется дескриптор:**

```
<script type="text/javascript">
```

```
</script>
```

Или сегодня можно использовать упрощенный вариант:

```
<script>
```

```
</script>
```

**Атрибуты дескриптора:**

- `type` — атрибут, пришедший на замену `language`; он сообщает браузеру, какой язык используется внутри дескрипторов;
- `src` — атрибут определяет URL внешнего исходного JavaScript-файла, сценарий которого прикрепляется к html-странице.

## Таким образом, варианты тега script:

1. Устаревший вариант:

```
<script language="javascript">...</script>
```

2. Действующие варианты:

```
<script type="text/javascript">...</script>
```

```
<script>...</script>
```

3. Вариант с прикрепленным файлом скрипта:

```
<script src="/jscripts/myscript.js">
```

```
</script>
```

## Итак, кратко резюмируем то, что необходимо знать о javascript:

- тег script обычно помещается в html-страницу в область head или body;
- этот тег указывает на то, что внутри находится сценарий — исполняемый код, в нашем случае скрипт на языке javascript;
- когда html-парсер браузера, отображая последовательно структуру html, доходит до тега script, то он передает инициативу

интерпретатору javascript;

- интерпретатор, в свою очередь, исполняет содержимое кода до закрывающего тега script , а затем опять передает управление

html-парсеру.

## Добавление javascript в html

Встраивание javascript в html происходит двумя основными способами, которые рассмотрим на примере:

**Пример 1.1:** Вывести в браузере сообщение «Это JavaScript!», используя скрипт на языке JavaScript.

Решение:

1. Встраивание JavaScript непосредственно в HTML-страницу:

- Создайте html-страницу со следующим кодом:

```
<html><head></head>
```

```
<body>
```

```
<!-- Сценарий -->
```

```
<script type="text/javascript">
```

```
    document.write("Это JavaScript!");
```


```
</script>
```

```
<!-- Конец сценария -->
```

```
<hr>
```

```
Это обычный HTML документ.
```

```
</body></html> •
```



Это JavaScript!

Это обычный HTML документ.

## 2. Размещение сценария во внешнем файле:

- Создайте две страницы: lab1.html и myscript.js. Расположите обе страницы в одном каталоге.
- В html-документе разместите код:

```
<html>
```

```
<head>
```

```
<!-- Прикрепление файла с кодом сценария -->
```

```
<script src="myscript.js"></script>
```

```
</head>
```

```
<body>
```

```
...
```

```
</body></html>
```

При прикреплении внешнего файла со скриптом тег script следует размещать в области head.

- В файле с расширением js (в данном конкретном примере — myscript.js) находится единственная строка — код для вывода сообщения:

```
document.write("Это JavaScript!")
```

- Откройте страницу lab1.html в браузере и посмотрите результат.

При прикреплении js-файла следует иметь в виду, что в html-файле надо указывать относительный путь к файлу со скриптом. Так, если файл со скриптом находится в каталоге jscripsts, то код будет: `script src="jscripsts/myscript.js"`.

- Функция `document.write()` используется для вывода информации на экран.
- Тогда как `document.writeln()` — используется для перевода на новую строку, если используется тег форматирования `pre`.
- `alert()` — это метод для вывода модального (диалогового) окна с сообщением

Выполните следующий пример, чтобы увидеть особенности работы метода `alert()`:

## Пример 1.2:

Поменяйте местами `alert` и `document.write`. Посмотрите на результат в браузере.

```
<script type="text/javascript">  
    alert ("Hello?");  
    document.write("Hello!");  
</script>
```

Важно: особенность модального окна `alert()` состоит в том, что пользователь не может продолжить работу, пока не щелкнет по кнопке окна

Обратите внимание, как работает `javascript`, помещенный в теги оформления шрифта:

### Пример 1.3:

Перенесите скрипт в BODY после тега H1. Посмотрите на результат.

```
<h1>
```

```
<script type="text/javascript">
```

```
  alert ("Hello?");
```

```
  document.write("Hello!");
```

```
</script>
```

```
</h1>
```

**Важно:** Пример показывает, что при использовании метода `write()` на странице выводится не просто текст, а html-код. То есть данный код может содержать теги html, которые будут преобразованы в соответствующее форматирование текста.



**Задание Js 1.1.** Вывести в окно браузера следующие данные: Ваше ФИО, возраст, хобби  
(каждое на новой строке)

**Задание Js 1.2.** Написать сценарий (javascript) для вывода двух строк текста, красной и синей.



Красная строка  
Синяя строка

Замечание:

Теги html для оформления текста цветом:

```
<font color="red">Красная строка</font>
```

**Задание Js 1.3.** Найдите и исправьте ошибки во фрагментах кода:

1. `document.whrit("Проблемы?");`
2. `alert>Hello);`

**Рассмотрим некоторые понятия, относящиеся к синтаксису языка:**

**Сценарий** — текст, состоящий из:

- операторов,
- блоков, т. е. взаимосвязанных наборов операторов, и
- комментариев.

**Операторы могут содержать:**

- переменные — могут изменять свое значение в программе,
- константы — не изменяют свое значение,
- выражения.

**Переменная** — это область памяти для хранения значений; для обращения к переменной используется ее имя (идентификатор). Кроме того, у переменной есть тип данных — это тип значения, которое принимает переменная.

**Идентификаторы (identifiers)** — имена переменных, методов и объектов:

- состоят из комбинации букв и цифр;
- должны начинаться либо с буквы, либо с символа подчеркивания;
- не должны содержать пробелов.

**Важно:** Язык JavaScript чувствителен к регистру:

//переменные различаются:

counter=1

Counter=1

**«Верблюжья нотация» в записи идентификаторов:**

Есть определенные устоявшиеся среди программистов правила для идентификаторов (имён) переменных, функций, массивов и классов. Рассмотрим их:

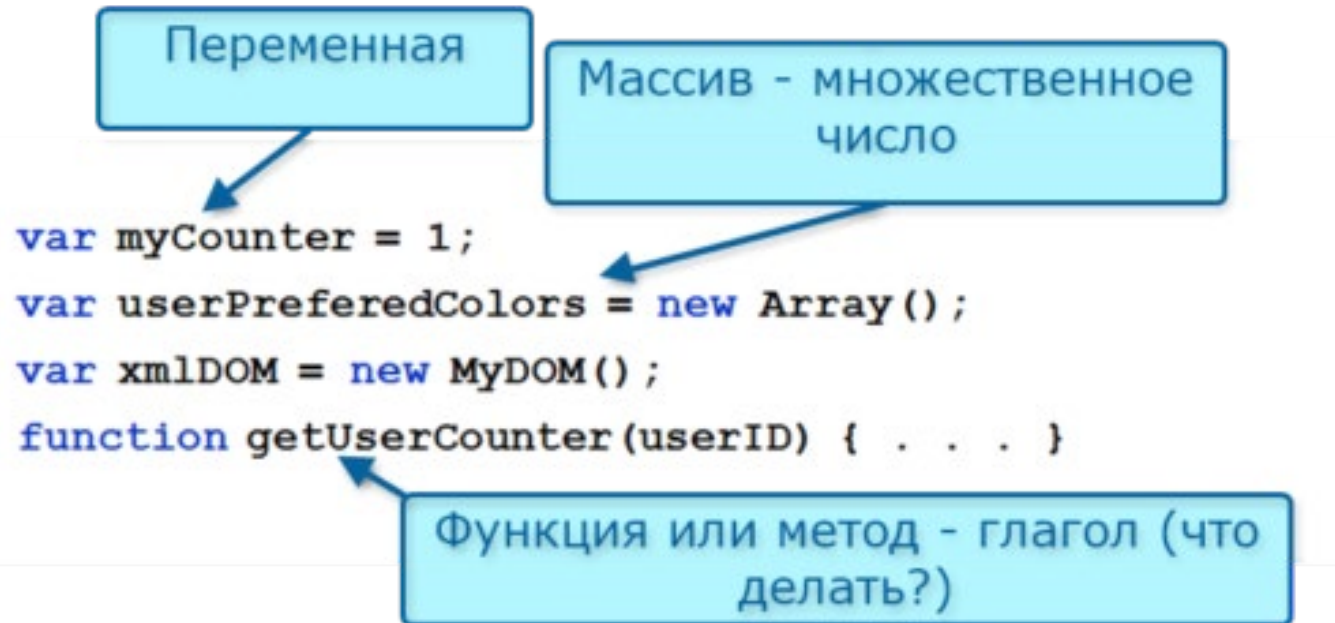
- **num\_docs** — знак подчеркивания между словами — хорошо, но есть способ лучше
  - **numDocs** — вот такой «верблюжий» стиль превосходно подходит для именованя переменных:
- ✓ все имена строчными буквами,
  - ✓ на стыке слов — большая буква,
  - ✓ переменные и свойства — называем именами существительными,
  - ✓ массивы и коллекции — называем существительными во множительном числе,
  - ✓ функции и методы — называем глаголами,
  - ✓ название классов — с заглавной буквы.

## Пример:

`var myCounter=1; // просто переменная`

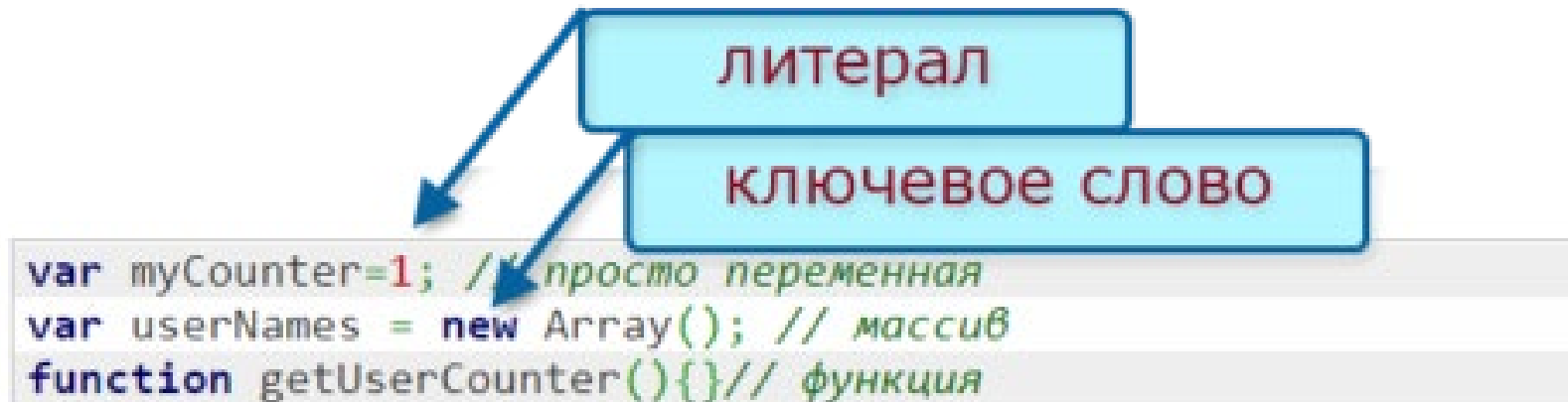
`var userNames = new Array(); // массив`

`function getUserCounter() {} // функция`



**Ключевые слова (keywords)** — предварительно определенные идентификаторы, составляющие основу языка программирования. Ключевые слова нельзя использовать для имен переменных, функций, объектов и методов.

**Литералы (literals)** — это постоянные значения JavaScript. Это значения, которые не изменяются во время выполнения сценария (целочисленные литералы, литералы с плавающей точкой, логические литералы (true и false), строковый литерал — это ноль и более символов, заключенных в двойные («») или одиночные (») кавычки).



## Правила оформления скрипта JavaScript

- каждый оператор JavaScript лучше начинать с новой строки;
- каждый оператор заканчивается точкой с запятой;

сегодня точка с запятой в конце оператора не обязательна, но если написать в строку несколько операторов (это тоже разрешается), то необходимо их разделить через ‘;’

### Такой код не работает:

```
a=5 document.write(a)
```

### Код работает верно:

// способ 1:

```
a=5
```

```
document.write(a)
```

// способ 2:

```
a=5; document.write(a);
```

// способ 3:

```
a=5;
```

```
document.write(a);
```

- блок — это набор операторов (составной оператор), заключенный в фигурные скобки { }.

```
{  
document.write(a);  
alert(b);  
}
```

## JavaScript комментарии

В JavaScript допустимы два вида операторов комментария:

1. // — одна строка символов, расположенная справа от этого оператора, считается комментарием;
2. /\*...\*/ — все, что заключено между /\* и \*/, считается комментарием; с помощью этого оператора можно

выделить несколько строк в качестве комментария.

```
// проверка
```

```
/* здесь может быть ошибка
```

```
a=5;  
document.write(a);  
*/
```

Второй способ комментирования обычно используется при поиске ошибок: тот блок сценария, в котором может находиться потенциальная ошибка, комментируется.

**Задание Js 1.4.** Исправьте ошибки во фрагменте кода:

```
alert("Hello World!"); / это однострочный комментарий
```

## **Объявление переменных в JavaScript и оператор присваивания**

**Переменная (variable)** — это имя, присваиваемое ячейке памяти компьютера, которая хранит определенные данные.

**JavaScript** — **нетипизированный язык**. Это значит, что переменные принимают тот тип данных, значение которого

в них присваивается. Напрямую задавать тип переменной не надо.

Объявление переменной происходит при помощи служебного слова javascript var:

```
var + имя переменной + ;
```

Пример объявления переменной:

```
var i;
```



**Оператор присваивания:**

```
i=0;
```

**Объявление переменной можно объединить с присваиванием:**

```
var + имя переменной + = + значение + ;
```

**Переменная всегда слева, справа – литерал (значение переменной).**

**Пример:**

```
var i=0;
```

**Таким образом, резюмируем. Существует три варианта объявления с присваиванием:**

**1. var hello = "привет";**

или

**2. var hello;**

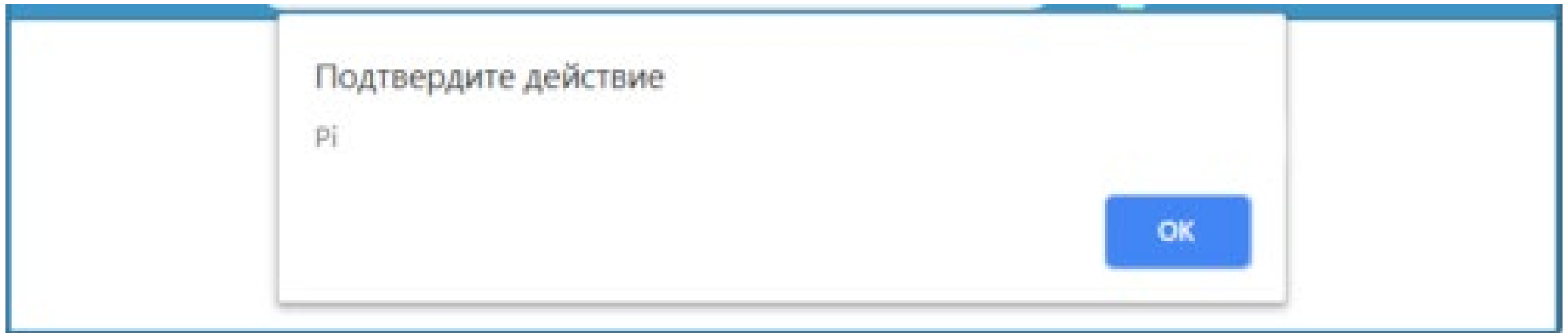
```
hello="привет";
```

или

**3. hello="привет";**

Объявление переменной при помощи служебного слова var можно опускать

**Пример 1.4:** Создать переменную с идентификатором `myVal` со строковым значением «Pi». Вывести значение переменной, используя модальное окно (метод `alert()`).



## Решение:

- Создайте html-страницу с тегом script, предназначенным для размещения дальнейшего кода:

```
<html>
```

```
<head></head>
```

```
<body>
```

```
<script type="text/javascript">
```

```
// будущий код javascript
```

```
</script>
```

```
</body>
```

```
</html>
```

- Добавьте код для объявления, инициализации переменной и вывода ее значения в предназначенное для

этого место:

```
var myVal; // объявляем переменную myVal
```

```
myVal = "Pi"; /* присваиваем myVal некоторое значение */
```

```
alert (myVal); // выводим значение
```

- Возможен также пример с необъявленной переменной:

```
myVal = "Pi"; /* присваиваем myVal некоторое значение */
```

```
alert (myVal); // выводим значение
```

- Запустите страницу в браузере и посмотрите на результат.

**Задание Js 1.5.** Объявите две переменные разными способами и присвойте им значения. Выведите на экран значения переменных при помощи метода alert()

**Задание Js 1.6.** Исправьте ошибки в правильности объявления локальных переменных во фрагменте кода:

```
String s = "String";
```

```
int a = 10;
```

```
long b = 25;
```

## Константы в JavaScript

### Объявление константы:

`const + имя константы + = + значение + ;`

`const MyX=2;`

Данные, присвоенные константе, в течение программы не меняются!

### JavaScript типы данных

Определение типа данных необходимо для установления операций, которые можно выполнить над переменными этого типа данных.

**Важно:** Но необходимо помнить, что в JavaScript типы переменных явно не указываются, т.к. это динамически типизированный, а не строго типизированный язык программирования.

Интерпретатор определяет тип переменной по правой части (по присвоенному ей значению).

Объявление локальных переменных осуществляется при помощи ключевого слова `var`.

## Тип данных

undefined type

## пример / объяснение

```
var x;  
alert (x);
```

значение, которое используется для переменных или свойств объекта, значения для которых не существует или оно не присвоено

Null type

```
var x=1;  
x=null //пустое значение
```

значение, указывающее на отсутствие объекта

Boolean type

логический (true или false)

```
var x=false;
```

String type

строковый

```
var x="Привет";
```

Number type

числовой

```
var x=3.14;  
var y=-567;
```

## Логический тип (boolean)

Пример использования логического типа:

```
var a = true;
```

```
var b = false;
```

```
c = a && b; // результат false
```

```
c = a || b; // результат true
```

```
c = !a; // результат false
```

**Javascript логические операторы:**

Оператор	Название	Пример
!	Отрицание (логическое НЕ)	!X
&&	логическое И	X && Y
	логическое ИЛИ	X  Y

## Задание Js 1.7.

Выполните задание по шагам:

- Создайте 3 переменные с использованием ключевого слова `var` с идентификаторами: `a`, `b`, `c`.
- Переменной `a` присвойте значение `false`.
- Переменной `b` присвойте значение `null`.
- Переменная `c` должна принимать значение `undefined`.
- Отобразите значение 3-х переменных последовательно в модальных окнах (то есть с помощью метода `alert()`).



## Строковый тип (string)

**Строка** — набор символов, обрамляется либо в `""`, либо в `"`

### Три способа создания строкового объекта:

1. `имя_переменной = new String("строковое_значение");`

`myString = new String ("Hello!");`

2. `имя_переменной = "строковое_значение";`

`myString = "Hello!";`

3. `var имя_переменной = "строковое_значение";`

`var myString = "Hello!";`

## Операции над строками

- Конкатенация объединение строк:

Пример:

```
var x="При";
```

```
var y="вет";
```

```
var s=x+y; //"Привет"
```

- Специальные символы:

\n — новая строка

\t — табуляция

### **Пример 1.5:**

Реализуйте приведенный ниже код, чтобы посмотреть, как работают специальные символы в javascript.

Запустите страницу в браузере:

```
alert("мама мыла раму");
```

```
alert("мама\n мыла\n раму");
```

### **Задание Js 1.8.**

Что должно быть в ответе на следующие присваивания?

"1"+2+3= ?

1+2+"3"= ?

### **Задание Js 1.9.**

Вывести в окно браузера при помощи метода alert() следующие данные: Ваше ФИО, возраст, хобби (каждое на новой строке: использовать специальные символы)

## Задание Js 1.10.

С помощью javascript метода `document.write()` вывести в окно браузера строку: Кто ты такой? (с пробелами между словами).

Последовательно выполните:

1. Создать 4 переменные с использованием ключевого слова `var` с именами `str1`, `str2`, `str3`, `concatenation`.
2. Переменной `str1` присвоить фразу 'Кто ', `str2` – 'ты ', `str3` – 'такой?'
3. Локальной переменной `concatenation` присвоить результат конкатенации 3-х строк: `str1`, `str2`, `str3`.
4. Вывести в документ содержимое переменной `concatenation`.

## Числовой тип (number)

В JavaScript существуют такие числовые типы:

- **int** — целое,
- **long** — длинное целое,
- **float** — вещественное.

**Но явное указание типов в коде при объявлении переменной не нужно!**

**Используется неявное объявление, без указания конкретного типа данных:**

```
var x = 5; // целое
```

```
var y = 5.6; // вещественное
```

**Другие примеры:**

```
var x = 5e3; // 5000
```

```
var y = 5e-3; // 0.005
```

**Префикс 16-ной системы в javascript 0x:**

```
var x = 0xFF; // 255
```

**Префикс 8-ной системы в javascript 0:**

```
var x = 045; // 37
```

```
var x = 0/0; // NaN - не число (not a number)
```

```
var x = 1/0; // Infinity (бесконечность)
```

**Задание Js 1.11.** Исправьте ошибки при объявлении локальных переменных во фрагментах кода:

1. String s = "String";

```
int a = 10;
```

```
long b = 25;
```

2. var name = "Меня зовут Вася ";

```
var 2b = 10;
```

```
Var _@c = 15;
```

```
alert(Name);
```

## Арифметические операторы javascript

Операторы предназначены для составления выражений.

Оператор применяется к одному или двум данным, которые в этом случае называются операндами.

**Например**, оператор сложения применяется к двум операндам ( $a + b$ ), а оператор логического отрицания — к одному операнду ( $\neg a$ ).

### Операторы присваивания:

- **= обычная операция присваивания;**

```
y = 5;
```

```
alert(y); // ВЫВОД 5
```

- **+=, -= присваивание со сложением или вычитанием;**

```
y = 5;
```

```
alert(y-=2); // ВЫВОД 3
```

- **\*=, /= присваивание с умножением или делением.**

```
y = 5;
```

```
alert(y*=2); // ВЫВОД 10
```

## Арифметические операторы:

- **сложение в javascript: +**

$X + Y;$

$y = 5;$

$x = y + 3; // \text{равно } 8$

- **вычитание в javascript: -**

$X - Y;$

- **умножение в javascript: \***

$X * Y;$

- **деление в javascript: /**

$X / Y;$

- **javascript остаток от деления или деление по модулю: %**

$X = 8;$

$Y = 5;$

$X \% Y; // 3$



- **javascript инкремент или увеличение на 1: ++**

X = 8;

X++; // 9

**/\* Префиксный инкремент выполняется перед использованием переменной, пример:\*/**

```
var number = 100;
```

```
++number; // примет значение 101
```

**/\* Постфиксный инкремент выполняется после использования переменной, пример:\*/**

```
var number = 100;
```

```
number++; // примет значение 100
```

```
alert(number); // выведет число 101
```

- **javascript декремент или уменьшение на 1: --**

Y=9;

Y--; // 8

## Задание Js 1.12.

Какие значения выведет в окно браузера следующий фрагмент кода?

```
var str = "20";  
  
var a = 5;  
  
document.write(str + a + "<br/>");  
  
document.write(str - a + "<br/>");  
  
document.write(str * "2" + "<br/>");  
  
document.write(str / 2 + "<br/>");
```

Имейте в виду, так как переменная `str` является строковым типом, то переменная `a` типа `Number` неявно преобразуется в строку и далее производится операция конкатенации. Но если операция сложения для строкового значения существует, то операции деления и вычитания для строк отсутствуют, соответственно, действия будут происходить с числами.

## **Задание Js 1.13.**

Необходимо написать сценарий, определяющий площадь прямоугольного треугольника по заданным катетам ( $S = ab/2$ ). Сценарий разместить в разделе `body` документа. С помощью скрипта вывести в окно браузера инкремент площади.

### **Алгоритм решения задачи на javascript:**

- Инициализация двух переменных.
- Вычисление площади.
- Вывод инкремента с использованием метода `write()`.