JavaScript

```
<!DOCTYPE HTML>
<html>
  <head> </head>
  <body>

    <script>


//тут ваша программа
//выполняется сразу после загрузки страницы
//это комментарий, не выполняется никак

    </script>

  </body>
</html>
```

JavaScript

Программа выполняется последовательно сверху вниз и слева направо, программа состоит из команд, которые разделяются точкой с запятой, каждая команда выполняет какое то действие:

Команда1;

Команда2; Команда 3;

Пример понятной программы:

Встать;
Подойти к холодильнику;
Открыть холодильник; Взять еду;
Закрыть холодильник;
Съесть еду;

JavaScript

Числа и строки с которыми может работать программа:

1. **Числа;**
   1
   1.5
   1e10

2. **Строки;**
   “Привет”, ‘Stroka’,”Тоже строка”

JavaScript

Познакомимся с несколькими командами для вывода данных:

1. Вывод данных в окошко:

alert(1);

2. Вывод прямо в документ (можно несколько значений через запятую)

document.write(1);

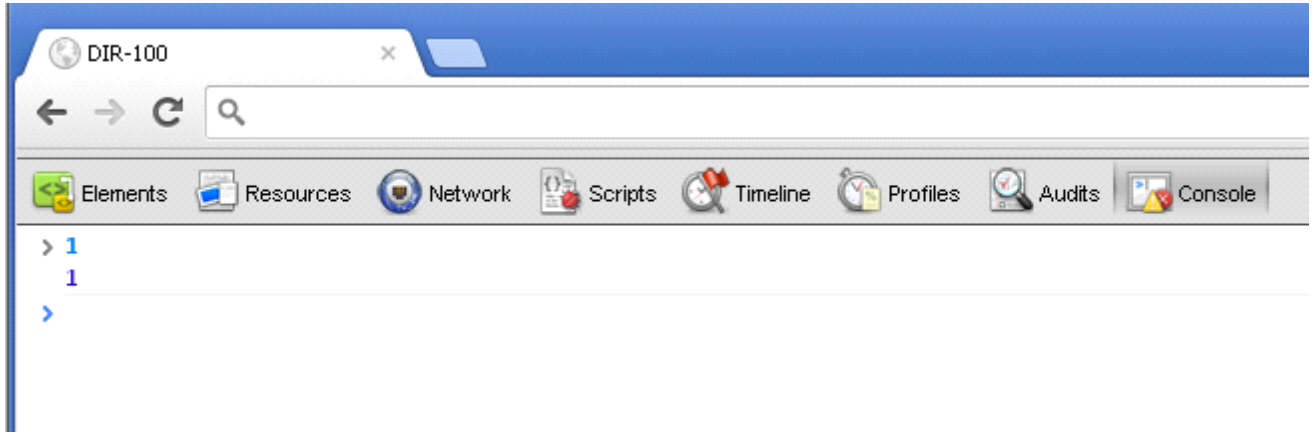
3. Вывод в консоль (F12):

console.log(1);

Эти команды выводят данные в скобках.

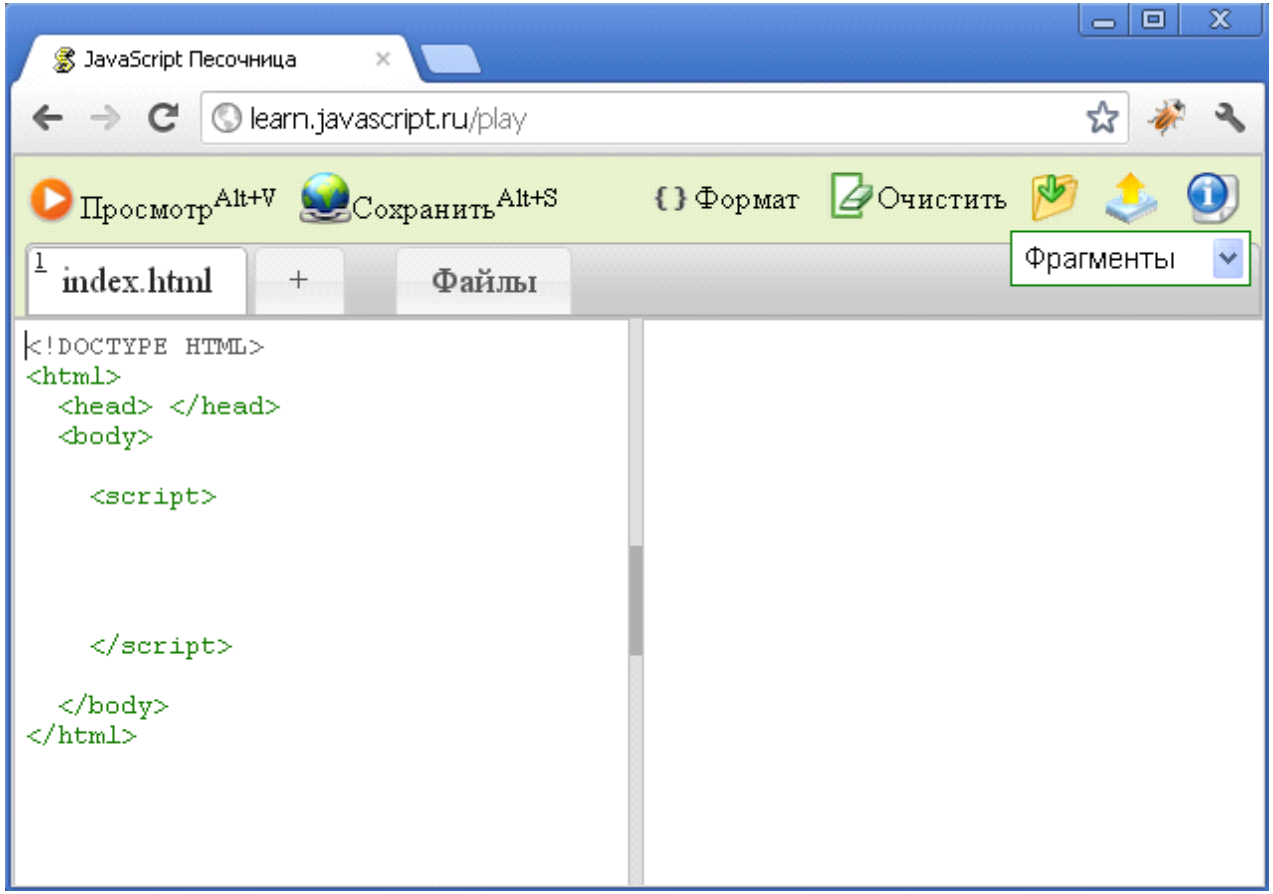
JavaScript

Все команды можно протестировать в консоли F12



JavaScript

Программы же можно тестировать в песочнице (console.log там не работает)



JavaScript

Обратите внимание на типичный вид команды:

```
alert(1);
```

Не забываем точку с запятой в конце каждой команды

В скобках данные с которыми работает команда

Название команды

JavaScript операторы

Операторы:

Для работы с данными существует вид команд – оператор:
+ - * / = () и т.п.

Например:
2+2
2*(3+5)
“Петя”+” Маша”

Это выражения, результатом выражения является число или строка, например если выполнить в консоли примеры выше то результатом будет:
4
16
“Петя Маша”

JavaScript переменные

Результаты вычислений и просто данные можно хранить в переменных.
Переменная, это ячейка памяти с именем, в которой хранится одно числовое или текстовое значение.

Переменная объявляется так:


```
var message;
```

Или можно сразу 2 так:

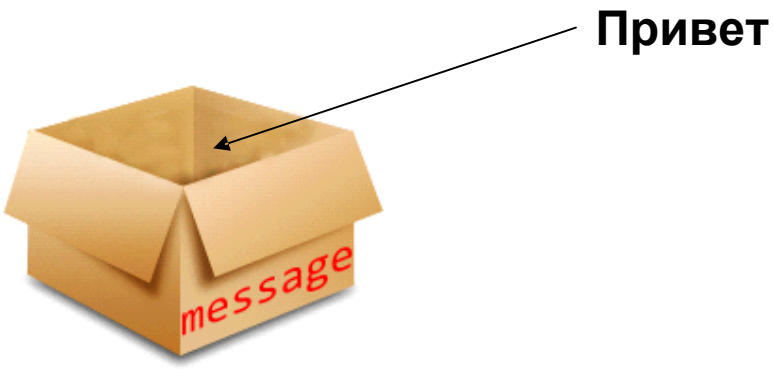
```
var A,B;
```

Переменная это как ячейка памяти в калькуляторе, можно хранить значения для вычислений.

JavaScript

Положить какое то значение в переменную можно оператором =
Например так:

message=“Привет”;



JavaScript

Еще примеры как заполнить ячейку памяти:

```
A=1;
B=2+2;
message=“hello”;
```

Оператор = работает как команда, поэтому его нужно завершать точной с запятой, слева от него всегда имя переменной в которую записываем данные, справа может быть либо данные либо выражение, которое сначала вычисляется, затем помещается в переменную.

ВНИМАНИЕ! Можно справа от = использовать имена других переменных, при этом из этих переменных извлекаются данные, и используются в выражении, например:

```
A=B+1;
```

Из переменной B извлекаются данные, затем вычисляется выражение прибавляя к нему единицу, затем, результат записывается в переменную A.

Как работает команда:

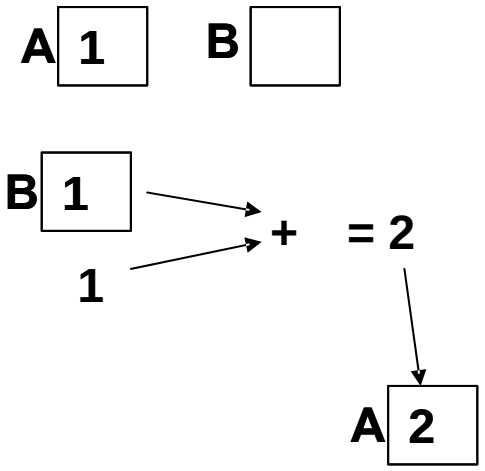
```
A=B+1;
```

1. Имеются 2 переменные A и B:

2. Из B извлекаются данные и вычисляется выражение B+1

3. Получается значение 2

4. Которое записывается в переменную A



JavaScript

Все переменные, которые вы используете в программе обязательно должны быть объявлены заранее, для предыдущего примера так:

```
var A,B=1;
```

Обратите внимание, что при объявлении переменных, можно сразу присваивать им значение!

JavaScript задание

Дано:
переменная **x = 1;**
переменная **y = 2;**

Задача 1 уровня:

Поменять содержимое переменных местами и вывести на экран (должно быть x=2 а y=1)

(подсказка, можно использовать дополнительные переменные)

Задача 2 уровня:

Поменять содержимое переменных местами не использую дополнительные переменные

Протестировать работу программы на разных исходных данных. Сделать ручной ввод данных.

JavaScript решение

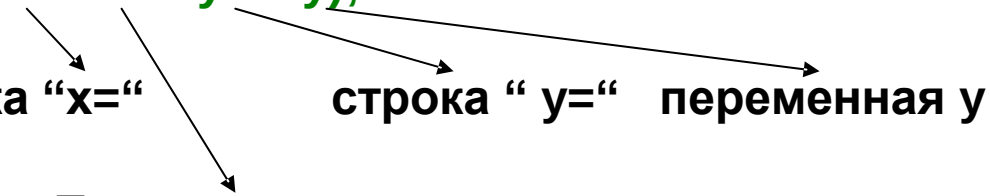
```
var x=1,y=2;
var tmp;

tmp=x;
x=y;
y=tmp;

alert("x="+x+" y="+y);
```

Обратите внимание на команду вывода результата:

```
alert("x="+x+" y="+y);
```



Строка “x=“

Переменная x

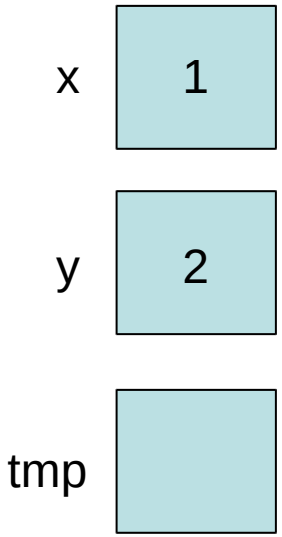
строка “ y=“

переменная y

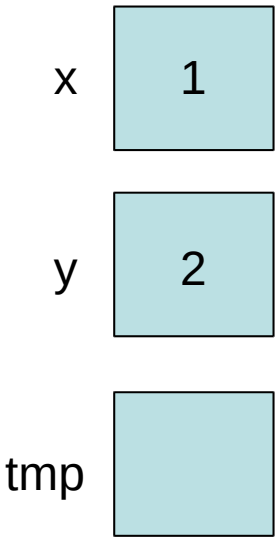
Распишем как работает программа:

```
var x=1,y=2;
var tmp;
```

Создаются 3 переменные, 2 уже со значениями:

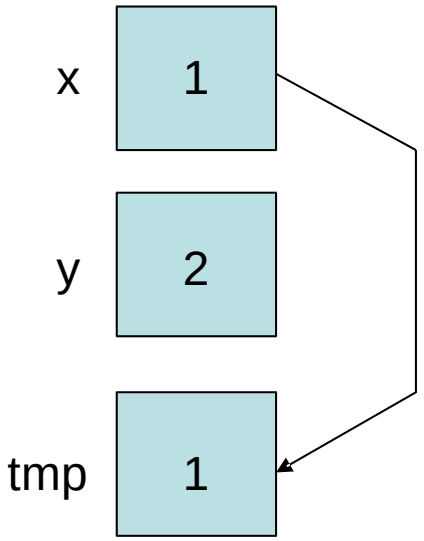


Было:

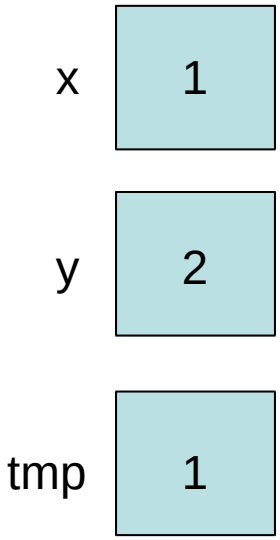


tmp=x;

Стало:

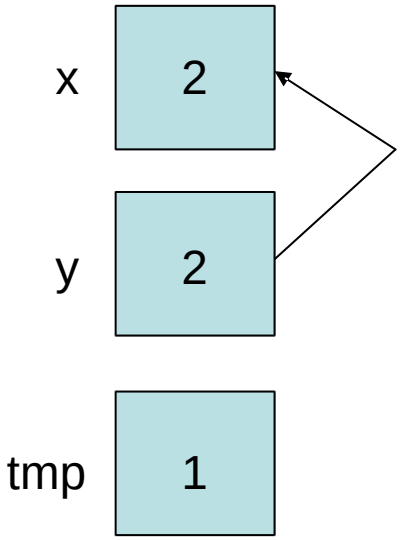


Было:

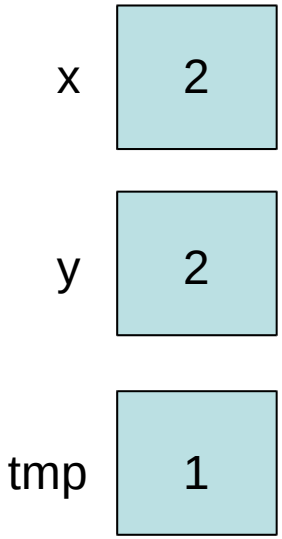


x=y;

Стало:

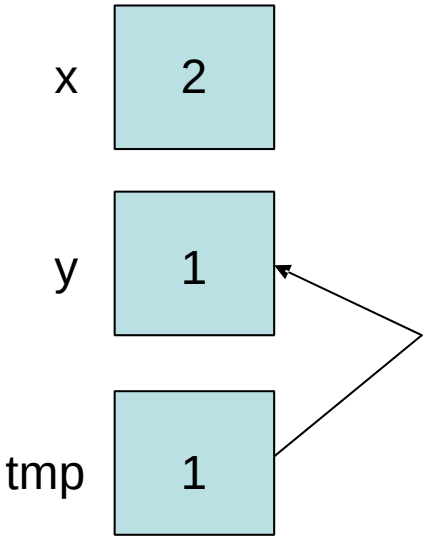


Было:



y=tmp;

Стало:



JavaScript

Операторы начинают выполняться сразу после загрузки страницы, чтобы выделить блок программы для выполнения «потом», ее нужно выделить в блок, называемый функцией:

Пример простой функции:

```
function showMessage() {   alert(“Привет всем присутствующим!”);}
```

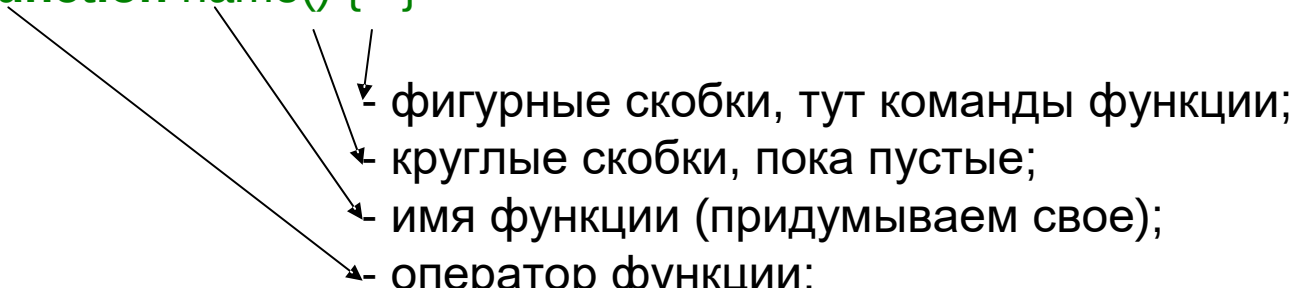
Вызов функции:

```
showMessage();
```

По сути, создавая функцию, мы как бы создаем новый оператор со своим названием, который можем использовать в программе как любой другой оператор.

Формат функции

```
function name() {   }
```



Пример тела функции:

```
{tmp=x; x=y; y=tmp;}
```

JavaScript задание

Составить программу:

1. **Создать свою функцию print_hello(); функция должна выводить в документ слово «Привет!»**

2. **Вызвать в основной программе функцию 2 раза, чтобы на экране было 2 слова привет;**

3. **Сделать так, чтобы каждое слово было с новой строки;**

JavaScript

```
function print_hello() {document.write("Привет!");}
print_hello();
document.write("<br>");
print_hello();
```

Обратите внимание, что теги в виде текста также можно выводить на экран, они будут исполняться браузером, как будто они встроены в html-странице.

JavaScript обработка нажатий

Клик по любому объекту на странице можно обработать в JS таким образом:

<img src=“image.jpg” onClick="программа;">
Или так
<div id=“my” onClick=“click_my();">

При нажатии кнопки будет выполнятся программа на JavaScript, удобно в качестве программы использовать один оператор функции, чтобы не перегружать выражение.

Можно сделать и кнопку так:
<input type="submit" value="OK" onClick="alert(1);">

JavaScript изменение координат

JS может изменять значения CSS-свойств, например, по нажатию на картинку, можно поменять ее координаты top и left, тогда картинка передвинется, делается это так:

```
<div id="raketa">Тут ракета</div>
```

Затем в программе к этому элементу можно обращаться и менять свойства так:

```
 <script>
      element = document.getElementById("raketa");
      element.style.top = "100px";
      element.style.left = "50px";
</script>
```

Примечание, атрибуты top и left должны быть явно заданы изначально. И можно обращаться только к элементам id. В значении свойств точку с запятой ставить не нужно.

Обратите внимание на новый оператор для выбора элемента для работы в программе:

```
element = document.getElementById("raketa");
```

- id элемента;
- название функции выбора;
- новая переменная, через которую потом можно будет работать в программе и обращаться к CSS-свойствам элемента id.

JavaScript

Затем можно работать со свойствами, например так:

element.style.**top** = "100px"; меняет позицию элемента
element.style.**visibility** = “hidden”; скрывает элемент
element.style.**color** = “red”; меняет цвет элемента

- значение свойства;
- имя свойства;

JavaScript

Пример:

```
<body>
<img id="atom" style="position: fixed; top: 10px; left:50px"
src="http://83.172.0.30/1.png" onClick="My();">
    <script>
          function My() {
             element = document.getElementById("atom");
            element.style.top = "100px";
            element.style.left = "50px";
                                    }
    </script>
  </body>
```

Зеленый – html;
Фиолетовый – CSS;
Красный – JS;

JavaScript

Значения для свойств стилей можно брать из переменных таким образом:

```
<body>
<img id="atom" style="position: fixed; top: 10px; left:50px"
src="http://83.172.0.30/1.png" onClick="My();">
   <script>
          var x=10;
          function My() {
          x=x+10;
           element = document.getElementById("atom");
          element.style.top = x+"px";
          element.style.left = "50px";
                                  }
   </script>
  </body>
```

создаем переменную - координату

увеличиваем значение координаты

так берем значение координаты и присваиваем его свойству top

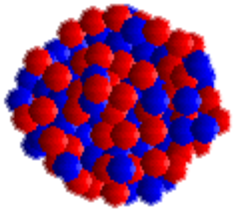
JavaScript задание шаблон

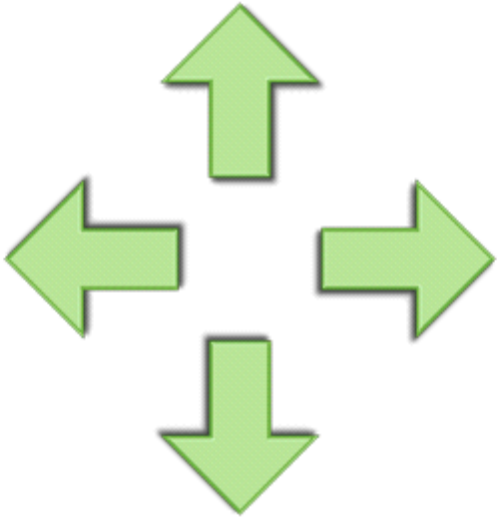
```
<!DOCTYPE html>
<html><head>    <meta charset="UTF-8"> <link rel="stylesheet" type="text/css"
href="style.css">
<title>Круглый блок</title>
 </head>
<body><img id="atom" style="position: fixed; top: 10px; left:120px"
src="http://83.172.0.30/1.png" onClick="My();">
<img id="atom" style="position: fixed; top: 300px; left:150px"
src="http://83.172.0.30/up.png"><img id="atom" style="position: fixed; top: 400px;
left:145px" src="http://83.172.0.30/down.png">
<img id="atom" style="position: fixed; top: 360px; left:45px"
src="http://83.172.0.30/left.png">
<img id="atom" style="position: fixed; top: 360px; left:235px"
src="http://83.172.0.30/right.png">

 <script> var x=10;
function My() {
x=x+10;
element = document.getElementById("atom");
element.style.top = x+"px";                    }
</script>
</body></html>
```

JavaScript задание

Сделать атом управляемым стрелками, он должен смещаться в соответсвующую сторону при нажатии на стрелку.





JavaScript управление с клавиатуры

```
<!DOCTYPE HTML>
<html>
  <head>  </head>
 <body id="main">
<input type="text" id="result">
   </body>

      <script>
//Получаем id-контейнера в котором будет обрабатывать нажатия
document.getElementById('main').onkeydown = test;
function test(e) {
  e = e || event;
  var evt = e.type;

  document.getElementById("result").value = e.keyCode;

}

    </script>
</html>
```

JavaScript массивы

Объявление массива:

```
var fruits = ["Яблоко", "Апельсин", "Слива"];
var N = [3, 6, 1]
```

Обращение к ячейкам массива:

```
fruits[1] = “Апельсин”;
N[0] = 3;
```

Нумерация в массиве начинается с нуля!

JavaScript задание

Сделать генератор ников используя текстовый массив
var A = ["ка", "ра", "ба", "с"]

Используйте выражение для генерации случайного числа, например от 1 до 5:
n1 = Math.floor(Math.random()*5)+1;

Сгенерируйте несколько случайных индексов массива и составьте из него результирующий ник, например:

result=A[n1]+A[n2]+A[n3]+A[n4];

Помните что нумерация в массиве с нуля!

JavaScript двумерные массивы

Объявление двумерного массива 2x2 (Массив в массиве):

```
var N = [[6, 3],[1,2]]
```

Обращение к ячейкам двумерного массива:

```
N[0][0] = 6;
N[1][0] = 1;
```

и т.п.

JavaScript условный оператор

Формат оператора if:

if (условие**) {**действие_если_истина**};**

Например:
if (5>2) {alert(“5>2”);}
if (5==5) {alert(“5=5”);}

Расширенный формат оператора if:
if (условие**) {**действие_если_истина;**} else {**действие_если_ложь;**}**

Например 2 не равно 1:
If (2!=1) {alert(“Истина”);} else {alert(“Ложь”);}

Широко используется составной операtр:
If (a>1**)** {Temp = a; a = b; b = temp;} **else** {alert(“a<1”);}

JavaScript логический тип данных

Разберем 3-ий вид данных JavaScript – логический:

Всего 2 значения переменной – **true** или **false**;

Операторы для сравнения:
> больше; **>=** больше или равно;
< меньше; **<=** меньше или равно;
== равно; **!=** не равно;

Например:
2 > 3 дает **false**;
1==1 получим **true**;
“Миша”==“Миша” получим **true**;

JavaScript

Можно объединять несколько условий в одном if при помощи оператора:

|| – или; **&&** - и;

Например

```
if ((a>1) && (a<100)) {alert("a находится между 1 и 100");}
```

JavaScript ввод данных

Ручной ввод данных в переменную можно сделать так:

```
var a;
a = prompt("Введите значение A");
document.write("a = ",a);
```

Проверка пароля так:

```
var pass;
pass = prompt("Введите пароль");
if (pass=="admin") {alert("Привет!")} else {alert("До свидания.")}
```

JavaScript цикл for

```
for (начало; условие; шаг) { // ... тело цикла ...}
```

Например:

```
var i;
for (i=1; i<=3; i++) {
document.write(i);
}
```

Выведет: 123

Внутри цикла переменная i меняет значение от 1 до 3 включительно, т.к. стоит условие <=, цикл удобно использовать для перебора значений, когда известно количество итераций.

JavaScript цикл for

Пример обратного отсчета цикла:

```
var i;
for (i=3; i>=1; i--) {
document.write(i);
}
```

Выведет: 321

JavaScript цикл for

Вывести значение переменной от 1 до 5 (12345) 3 раза,
должно получиться так:
1 12345
2 12345
3 12345

Подсказки:
Перенос строки можно сделать так:
**document.write(i,”
”);**

Пробел добавить можно так:
document.write(i,” ”);

JavaScript цикл for

```
var i,j;
  for (j=1;j<=3;j++)
            {   document.write(j," ");
                for(i=1;i<=5;i++)  document.write(i);
                document.write("<br>");
            }
```

JavaScript цикл while

```
while (условие)
{  // код, тело цикла}
```

Например:

```
var i = 1;
while (i <= 3)
{ document.write(i);
   i = i + 1;
}
```

Выведет: 123

Однако, этот цикл не очень удобен для перечислений, лучше использовать цикл for, этот же цикл удобно использовать для бесконечных циклов с выходом по условию, например так:

JavaScript цикл while

```
var pass = “”;
while (pass != “admin”)
{
   pass=prompt(“Введите пароль!”);
}
document.write(“Добро пожаловать!”);
```

JavaScript цикл и массивы

Циклы удобно использовать для перебора значений массива:

```
var A = [3, 2, 1];
for(var i=0;i<A.length;i++) document.write(A[i],”<br>”);
```

Выводит:
3
2
1

Используется новая функция определения длины массива:
A.length

Хотя просто распечатать значение массива можно так:
document.write(A);

JavaScript цикл и массивы задание

Даны массив:
var A = [1, 2, 3, 4, 5, 0, 0, 0, 0, 0];

Сделать так, чтобы первые 5 значений массива из 10 элементов заполнили полностью массив продублировав каждый элемент, в результате должно получится так:

[1, 1, 2, 2, 3, 3, 4, 4, 5, 5];

Если чувствуете силы, попробуйте сделать упражнение без промежуточных массивов.

JavaScript цикл и массивы решение

```
var A = [1, 2, 3, 4, 5, 0, 0, 0, 0, 0];
var i;
for (i=4; i>=0; i--) {
  A[i*2]=A[i];
  A[i*2+1]=A[i];
}
document.write(A);
```

JavaScript ввод-вывод через формы

Формы для ввода-вывода данных рисуются так:

```
<input type="text" id="result">
<input type="text" id="input">
```

где id – обозначение формы по которой будем обращаться так:
Взять значение из поля:

```
a = document.getElementById(“input”).value;
```

Записать значение в поле:

```
document.getElementById(“input”).value = 12345;
```

JavaScript пример ввода вывода

```
<input type="password" id="input" ><br>
<input type="text"  id="result" disabled >
<input type="submit" value="OK" onClick="go();">


<script>
function go(){
a = document.getElementById("input").value;
document.getElementById("result").value = a;
    }
</script>
```

Здесь выводится поле ввода пароля, и поле для вывода результата по нажатию кнопки ОК, тип **password** позволяет делать поля для паролей, а свойство **disabled** не позволяет изменять поля для вывода данных.

JavaScript

Теперь сделаем форму ввода пароля с подтверждением, если пароль введен второй раз верно форма становится зеленой, иначе красной:

```
  Введите пароль:<br>
  <input onkeyup="check();" type="password"  id="pass1"><br>
  Повторите пароль:<br>
  <input onkeyup="check();" type="password"  id="pass2">
```

И напишем обработчик событий нажатия клавиш:

```
<script>
  function check(){
  pass1 = document.getElementById("pass1").value;
  pass2 =  document.getElementById("pass2").value;
  if (pass1==pass2) {
    document.getElementById("pass2").style.background="green";
    document.getElementById("pass1").style.background="green“;
}
    else {
      document.getElementById("pass1").style.background="red";
      document.getElementById("pass2").style.background="red";
    }
       }
  </script>
```

JavaScript

Информация по работе с узлами:

https://learn.javascript.ru/modifying-document

JavaScript

Можно динамически управлять содержимым страницы, например создать любой текст внутри блока с id=“area”:

```
 <button onclick="add();">Добавить 1 поле ввода:</button><br>
<span id="area"></span>
  <script>
  function add(){
document.getElementById('area').innerHTML ='<input type="text">';
        }
</script>
```

JavaScript

Можно создавать много элементов, прикрепляя его к любому другому:

```
 <button onclick="add();">Добавить поля ввода:</button><br>
<span id="area"></span>
  <script>
  function add(){
    var div = document.createElement('div');
    div.innerHTML = "<input type='text'>";
    area.appendChild(div);
       }
  </script>
```

Шпаргалка по свойствам CSS

width:1200px; - фиксированная ширина блока;

max-width:1200px; - максимальная ширина резинового блока;

height:50px; - фиксированная высота блока;

background:green; - цвет блока зеленый;

border:2px solid red; - граница блока 2 пикселя сплошная красная;

float:right; - блок плавающий вправо;

padding:5px; - внутренний отступ;

padding-left:5px; - внутренний отступ слева;

margin:150px; - внешний отступ;

margin-right:150px; - внешний отступ справа;

text-indent:2em; - отступ абзаца;

text-align: justify; - выравнивание текста по ширине;

Шпаргалка по свойствам CSS

Города при наведении курсора должны подсвечиваться (изменяться фон и становиться жирным шрифт)

Подсказка:

Фон – **#6ca99e**

Заголовок – **#98bfde**

Подвал - **#333333**

Символ города - •

Подсветка по наведению на блок – **#block:hover {**

Шрифт нормальный 14-размер фонт-arial:

font:normal 14pt arial;

Просто жирный шрифт:

font-weight:bold;

Картинка (html):

<img src="map.png">

Дополнительно

Примеры элементов можно взять тут:

http://www.weblabla.ru/examples/html/examples_main.html