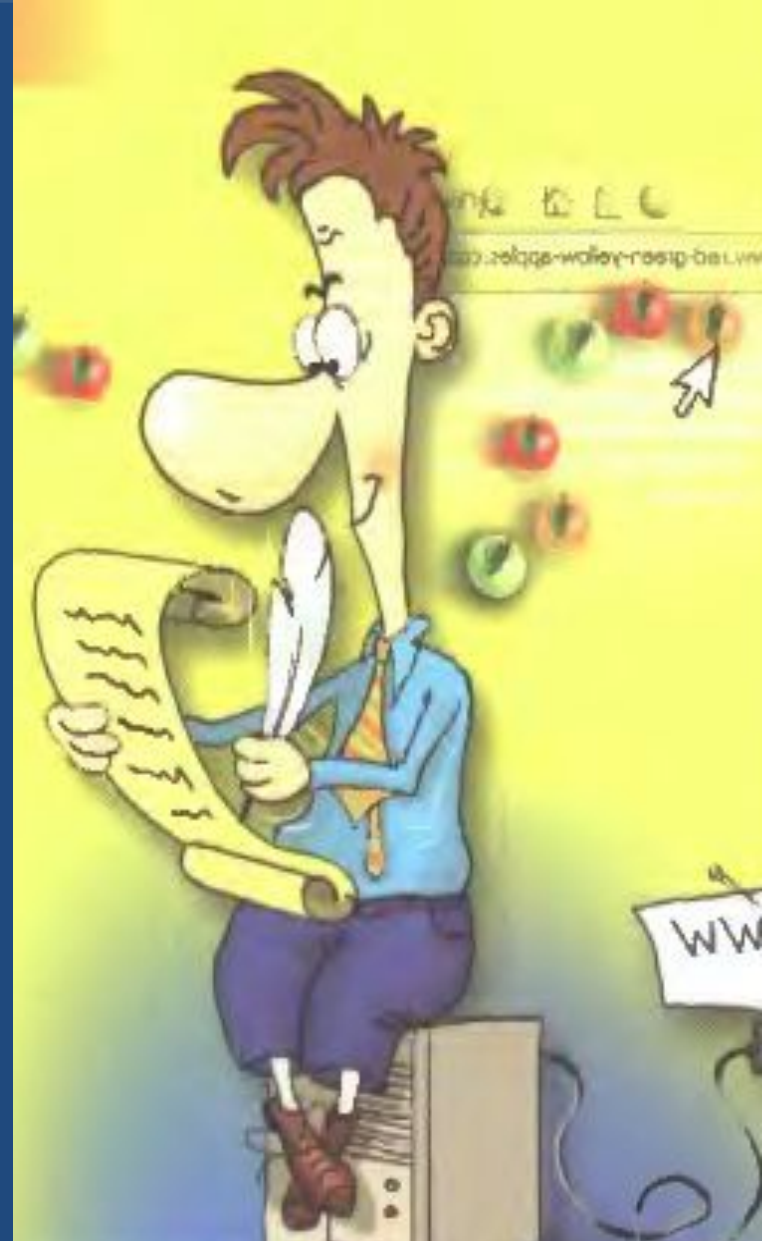


WEB – ПРОГРАММИРОВАН

Уроки практического
программирования

УРОК 2



Переменные и операторы

Переменные

ПЕРЕМЕННАЯ – ?...

контейнер для хранения
данных.

Правила записи имен переменных:

Переменная имеет имя – это....?

- последовательность букв, цифр и
- символа подчеркивания
- без пробелов,
- без знаков препинания,
начинается обязательно с буквы

Переменные

Правильные имена:

НЕ правильные

Java Script является регистрозависимым языком.

- Это значит, что изменение регистра символа (с прописной на строчную и наоборот) в имени переменной приводит к другой переменной.

Например:

MyField myField MyField

Переменные

Инициализация переменных в коде программы осуществляется с помощью служебного слова **VAR**, причем так как Java Script является слаботипизированным языком – **объявление переменной и её типа является не обязательным**.

- Переменной присваивается строковое значение:

MyGrup = "P2012"

- Инициализация переменной без присвоения значения:

VAR MyGrup

- Инициализация переменной с одновременным присвоением значения: **VAR MyGrup = "P2012"**

- Одновременная инициализация нескольких переменных с присвоением значения:

VAR MyGrup = "P2012", MySpetc = "230105.51"

Область действия

□ Переменной которые созданы в программе с помощью оператора присваивания с использованием ключевого слова VAR или без него, являются **ГЛОБАЛЬНЫМИ**.

□ Это значит, что переменные **доступны всюду** в этой программе, а также в вызываемых программах из других файлов. Эти же переменные **доступны**

□ **внутри** переменные объявленные внутри кода функции являются **ЛОКАЛЬНЫМИ** и не доступны из внешнего кода программы.

Область действия



- Область видимости переменной;
- Область доступности переменной;
- Область действия переменной.

**ЭКВИВАЛЕНТН
ые ПОНЯТИЯ**

Кроме них еще существует термин
– **ВРЕМЯ ЖИЗНИ ПЕРЕМЕННОЙ.**

В Java Script время жизни переменной

определяется интервалом времени от загрузки до выгрузки программы из памяти компьютера.

Так, если программа (сценарий) записаны в HTML – коде web – страницы, то после выгрузки весь сценарий вместе с определенными в нем переменными прекращает активное существование.

ОПЕРАТОРЫ

Операторы предназначены для составления выражения. Оператор применяется к одному или нескольким данным, которые в этом случае

В JavaScript допустимы два вида операторов комментария:

// - одна строка символов

/* ... */ - все что заключено между /* и */; несколько строк комментария

АРИФМЕТИЧЕСКИЕ

Оператор	Название	Пример
+	Сложение	$X + Y$
-	Вычитание	$X - Y$
*	Умножение	$X * Y$
/	Деление	X / Y
%	Деление по модулю	$X \% Y$
++	Увеличение на 1	$X++$
--	Уменьшение на 1	$Y--$

АРИФМЕТИЧЕСКИЕ

Если один операнд **строкового** типа, а другой **логического**, то в случае сложения интерпретатор переведет оба операнда в строковый тип и возвратит строку – результат конкатенации строк, в случае же других

арифметических операторов он переведет оба операнда в числовой тип:

«программист» + true // «программист true»

«5» + true // «5true»

«программист» * true // NaN «не число»

5 * true // число 5

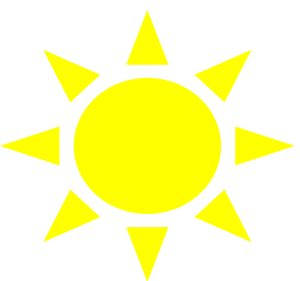
5 * false // число 0

5 / true // число 5

Дополнительные ОПЕРАТОРЫ

присваивания

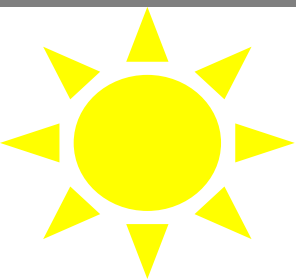
Оператор	Пример	Эквивалентное выражение
$+=$	$X+=Y$	$X=X+Y$
$-=$	$X-=Y$	$X=X-Y$
$*=$	$X*=Y$	$X=X*Y$
$/=$	$X/=Y$	$X=X/Y$
$\%=$	$X\%=Y$	$X=X\%Y$



Дополнительные операторы присваивания просто сокращают запись кода программы. По началу можно пользоваться обычным оператором

ОПЕРАТОРЫ СРАВНЕНИЯ

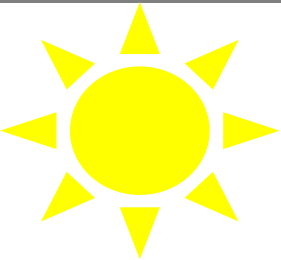
ОПЕРАТОР	НАЗВАНИЕ	ПРИМЕР
<code>==</code>	Равно	<code>X==Y</code>
<code>!=</code>	НЕ равно	<code>X!=Y</code>
<code>></code>	Больше чем	<code>X>Y</code>
<code>>=</code>	Больше или равно (не меньше)	<code>X>=Y</code>
<code><</code>	Меньше, чем	<code>X<Y</code>
<code><=</code>	Меньше или рвно (не больше)	<code>X<=Y</code>



Сравнивать можно числа, логические значения и строки. Сравнение строк происходит путем сравнения ASCII кодов.

ЛОГИЧЕСКИЕ ОПЕРАТОРЫ

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

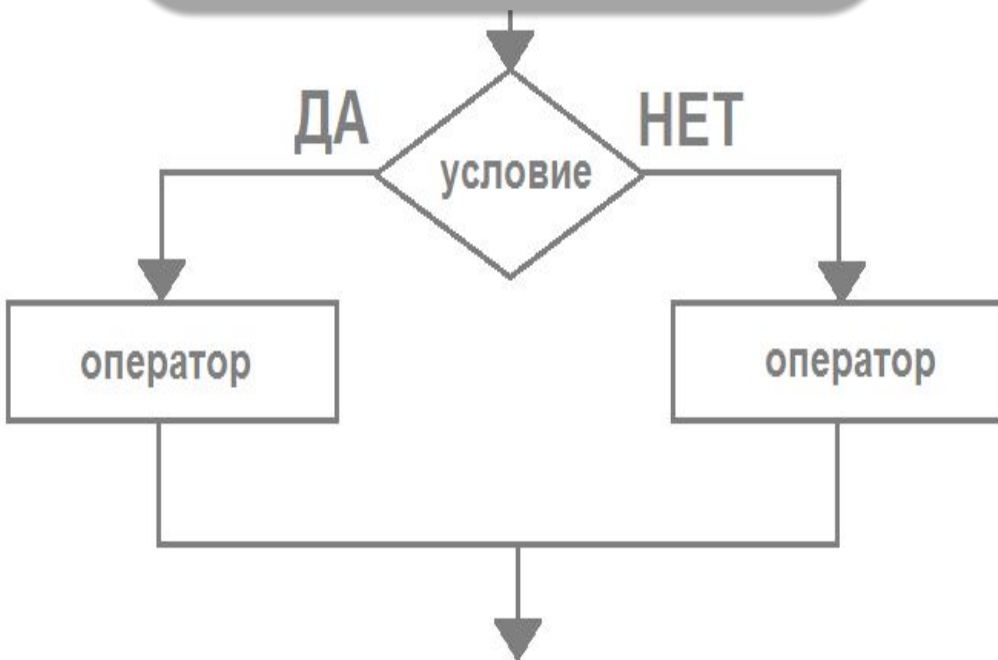


Оператор **ОТРИЦАНИЯ** применяется к одному операнду, операторы **И**, **ИЛИ** к двум операндам.

Логические операторы лучше **НЕ ПРИМЕНЯТЬ** к данным **НЕ ЛОГИЧЕСКОГО ТИПА** или к данным

ОПЕРАТОРЫ условного

Оператор **if** (условие)



```
{  
  блок операторов
```

```
}
```

Else

```
{
```

```
  блок операторов
```

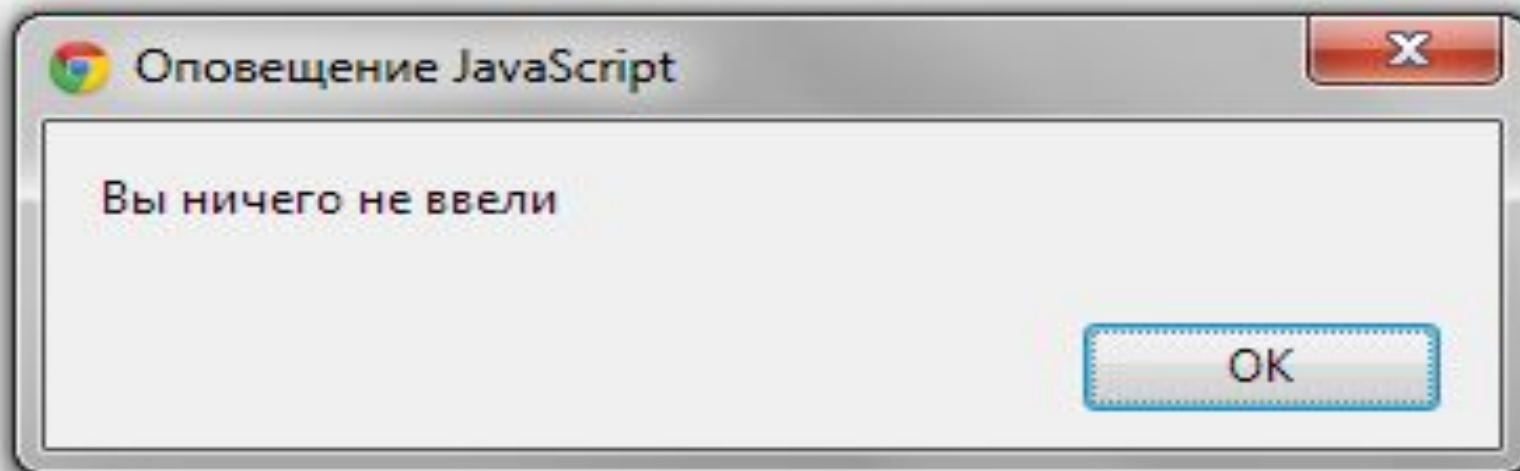
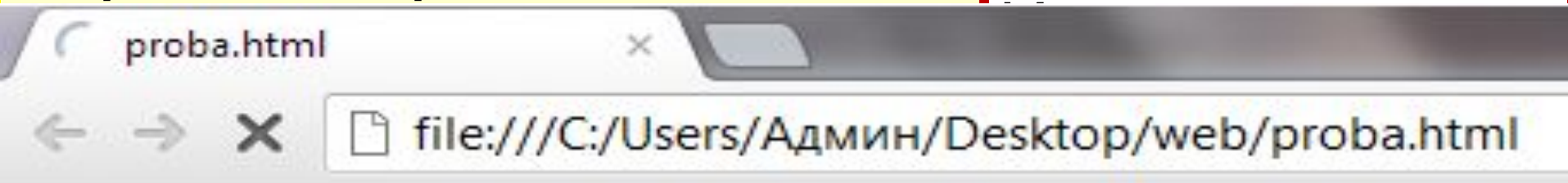
УСЛОВИЕ – ВЫРАЖЕНИЕ ЛОГИЧЕСКОГО ТИПА!

Однако в Java Script в качестве условия может выступать числовое или строковое выражение. В первом случае, условие будет считаться истинным если значение равно числу отличному от нуля, во втором случае строка является

ОПЕРАТОРЫ условного

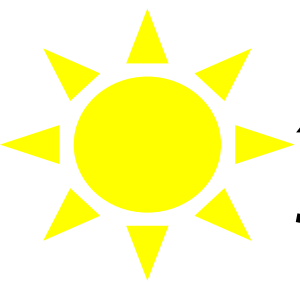
Оператор **if** перехода:

Необходимо проверить ввел ли пользователь данные. Предположим, данные введенные пользователем должны сохраняться в переменной X.



ОПЕРАТОРЫ условного

```
proba.html — Блокнот
Файл  П_равка  Формат  В_ид  С_правка
<HTML>
<Script>
var x
if (!x)alert("Вы ничего не ввели")
</Script>
</HTML>
```



Так как в данном примере, ветвление является не полным и содержит всего лишь один оператор, то фигурные скобки можно опустить.

ОПЕРАТОРЫ условного перехода

Оператор Switch

перехода

```
switch (выражение) {
```

Case вариант 1:

Код

[break]

Case вариант 2:

Код

[break]

[default:

код]



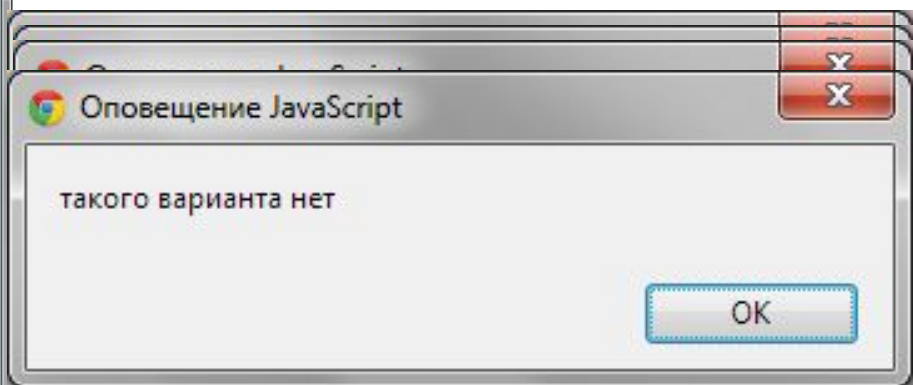
Ключевые слова *default* и *break* могут быть опущены. Если *Break* указан, то выполнение всех остальных операторов не производится. Блок операторов *Default* выполняется если искомого значения не

ОПЕРАТОРЫ условного

Оператор **switch** **перехода**

Пример 1:

```
proba.html — Блокнот
Файл Правка Формат Вид Справка
<HTML>
<Script>
var x=4
switch (x){
  case 1: alert("вы выбрали 1-й вариант")
          brake
  case 2: alert("вы выбрали 2-й вариант")
          brake
  case 3: alert("вы выбрали 3-й вариант")
          brake
  default:alert("такого варианта нет")
}
</Script>
</HTML>
```

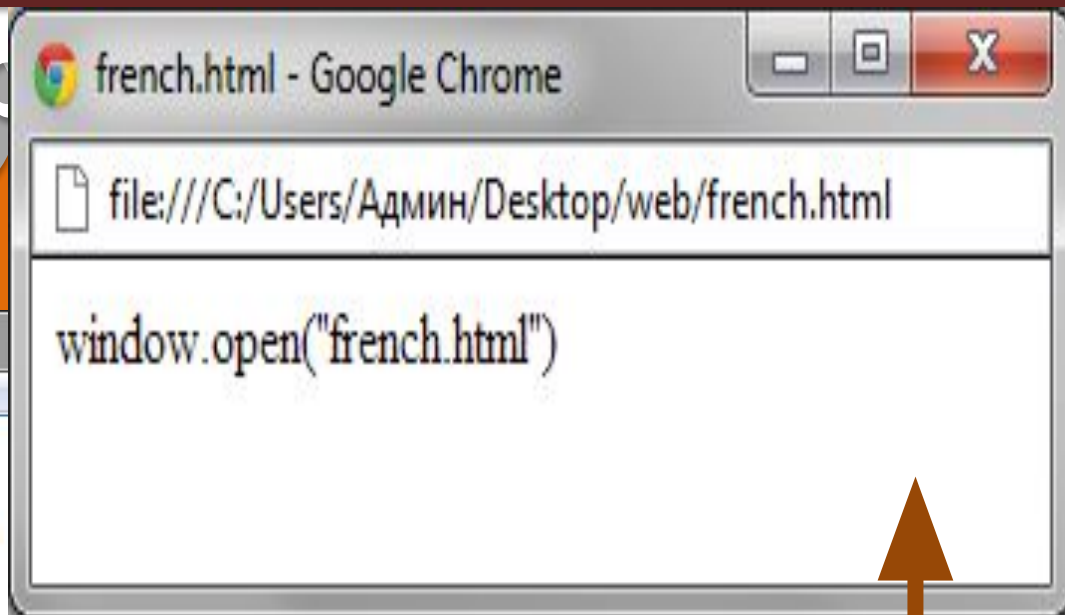


ОПЕРАТОРЫ условного

Оператор switch

Пример 2:

```
proba.html — Блокнот
Файл  Правка  Формат  Вид  Справка
<HTML>
<Script>
var xlang="французский"
switch (xlang){
  case "английский": window.open("eng1.html")
                    brake
  case "немецкий":  window.open("german.html")
                    brake
  case "французский": window.open("french.html")
                    brake
  default:alert("такого варианта нет")
}
</Script>
</HTML>
```



window.open("имя файла") – открывает новое окно браузера и загружает в него

ОПЕРАТОРЫ ЦИКЛА

Начальное выражение определяет значение счетчика в начале выполнения цикла. Начальное выражение выполняется параметр – **условие**, представляет собой условие продолжения выполнения

Третий параметр – представляет собой **выражение**, которое выполняется после выполнения каждой итерации.

FOR ([начальное выражение]; [условие]: [выражение обновления])

Квадратные скобки в записи параметров цикла указывают

на их не обязательность:

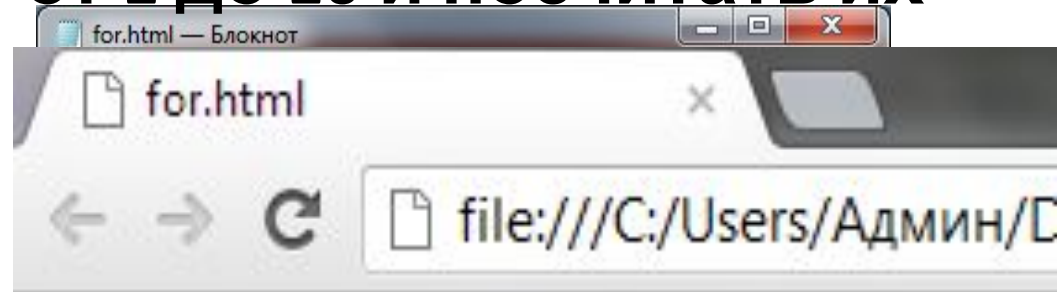
{
тело цикла

ОПЕРАТОРЫ ЦИКЛА

Оператор FOR

Пример 1

Необходимо вывести на экран все нечетные числа от 1 до 10 и посчитать их



1

3

5

7

9

сумма нечетных чисел = 25

ОПЕРАТОРЫ ЦИКЛА

Оператор FOR

Пример 2

Необходимо вывести на экран
все значение X в степени

1,2,...Y:

```
Файл Правка Формат Вид Справка
<HTML>
<Script>
//возведение числа-x в степень-y
x=3
var z = x
y=8
for(i = 1;i <= y;i++)
{
document.write(x+" в степени "+i+" = ")
document.write(z)
document.write("<br>")
z = z * x
}

</Script>
</HTML>
```



3 в степени 1 = 3
3 в степени 2 = 9
3 в степени 3 = 27
3 в степени 4 = 81
3 в степени 5 = 243
3 в степени 6 = 729
3 в степени 7 = 2187
3 в степени 8 = 6561

ОПЕРАТОРЫ ЦИКЛА

Оператор FOR

Пример 2

Необходимо вывести на экран факториал числа $n! = 1 * 2 * 3 * \dots * n$:

например : $1 * 2 * 3 * 4 = 24$

```
Файл  Правка  Формат  Вид  Справка
<HTML>|
<Script>
//вычисление факториала  числа
n=4
var z = 1
if (n>1)
{
  for(i = 1;i <= n;i++)
  {
    z = z * i

    if (i<n) document.write(i+"*")
    else document.write(i)

  }
}
document.write("="+z)
</Script>
</HTML>
```

for_факториал_числа.htm x
file:///C:/Users/Ад

$1 * 2 * 3 * 4 = 24$

ОПЕРАТОРЫ ЦИКЛА

Внутри тела цикла можно использовать оператор прерывания цикла (**break**) и оператор прерывания текущей итерации цикла (**continue**).

Оператор **break**

Прерывает
выполнение
оператора **цикла** и
ВЫХОДИТ из цикла.

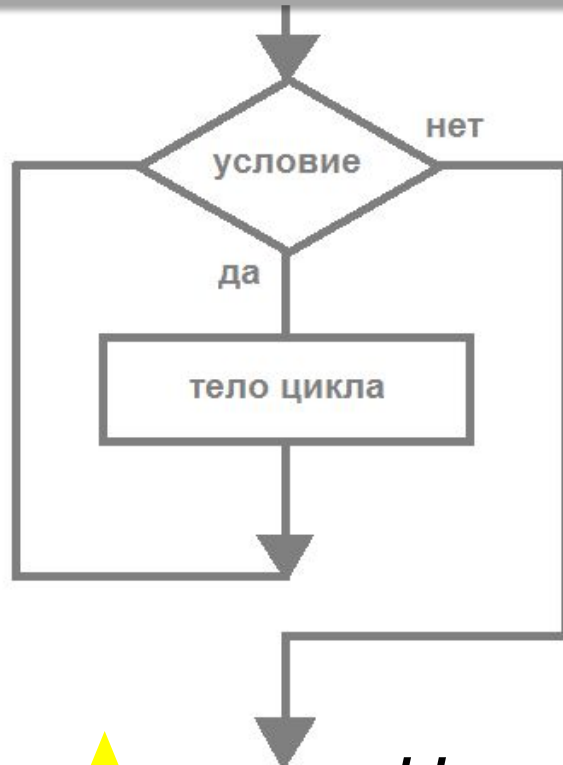
Оператор **continue**

Прерывает
выполнение
итерации цикла и
переходит к
следующей

ОПЕРАТОРЫ ЦИКЛА

Оператор while

ФОРМАТ



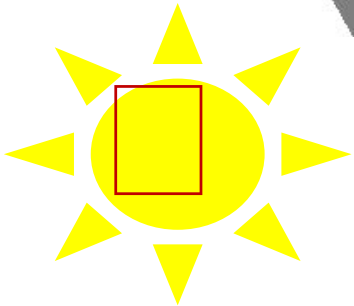
while

(условие)

{

Тело цикла

Цикл с предусловием или цикл «ПОКА»
Пока условие истинно выполняются
операторы тела цикла !

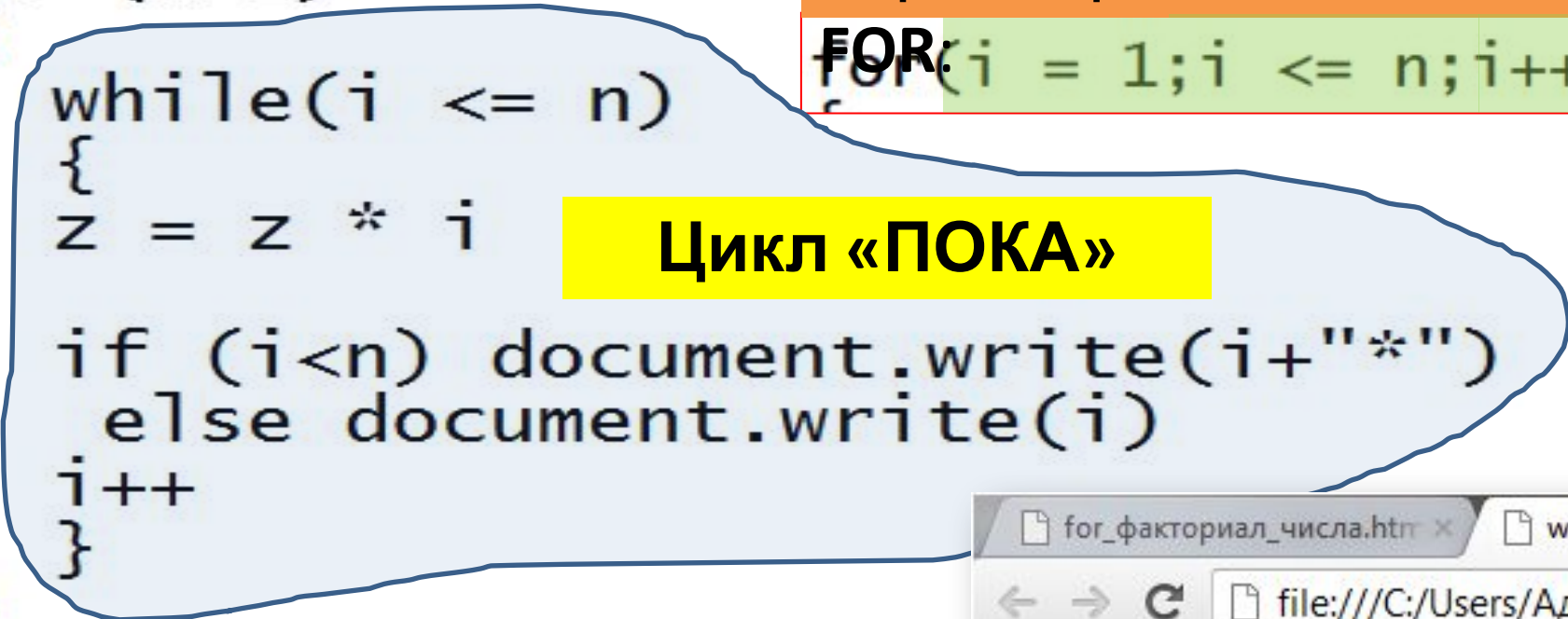



```
<HTML>
<Script>
//вычисление факториала числа
n=8
```

```
var z = 1, i = 1
if (n>1)
{
```

Сравните с параметрами цикла

```
FOR: for(i = 1; i <= n; i++)
```

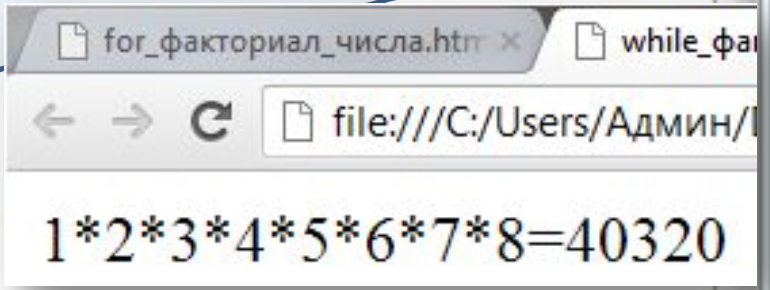


Цикл «ПОКА»

```
while(i <= n)
{
z = z * i

if (i<n) document.write(i+"*")
else document.write(i)
i++
}
}
```

```
document.write("="+z)
</Script>
</HTML>
```



1*2*3*4*5*6*7*8=40320

ОПЕРАТОРЫ ЦИКЛА

Оператор
do...while

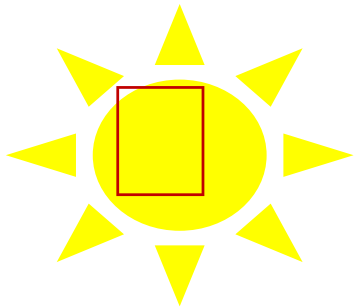
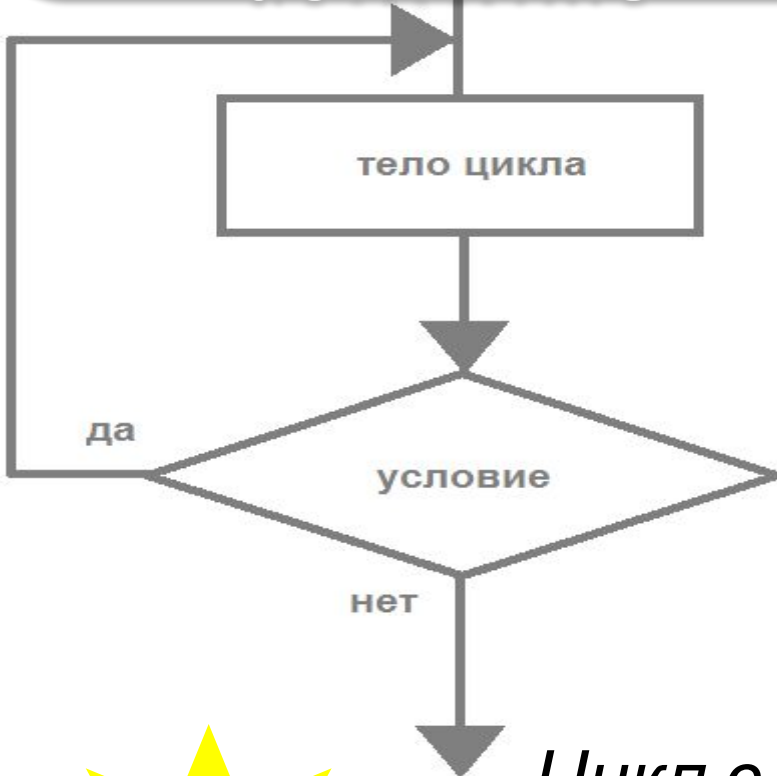
ФОРМАТ ЗАПИСИ:

```
Do {
```

```
Тело цикла
```

```
}
```

```
while
```



Цикл с постусловием или цикл «ДО»
(условие)
**ДО тех пор, пока условие истинно
выполняются операторы тела цикла !**

```
<HTML>
<Script>
//вычисление факториала
n=5|
var z = 1, i = 1
if (n>1)
{
do
{
z = z * i
if (i<n) document.write(i+"*")
else document.write(i)
i++
}
while(i <= n)
}
document.write("="+z)
</Script>
</HTML>
```

$$1*2*3*4*5=120$$

Цикл «ДО»