Мировые информационные ресурсы

Преподаватель:

Попков Сергей Игоревич,
аспирант,
заведующий лабораторией

(rsteach0@gmail.com)

Лекция 2. Краткое введение в эволюцию языков программирования; основы JavaScript

Данный курс не изучает основы общего (стационарного) программирования

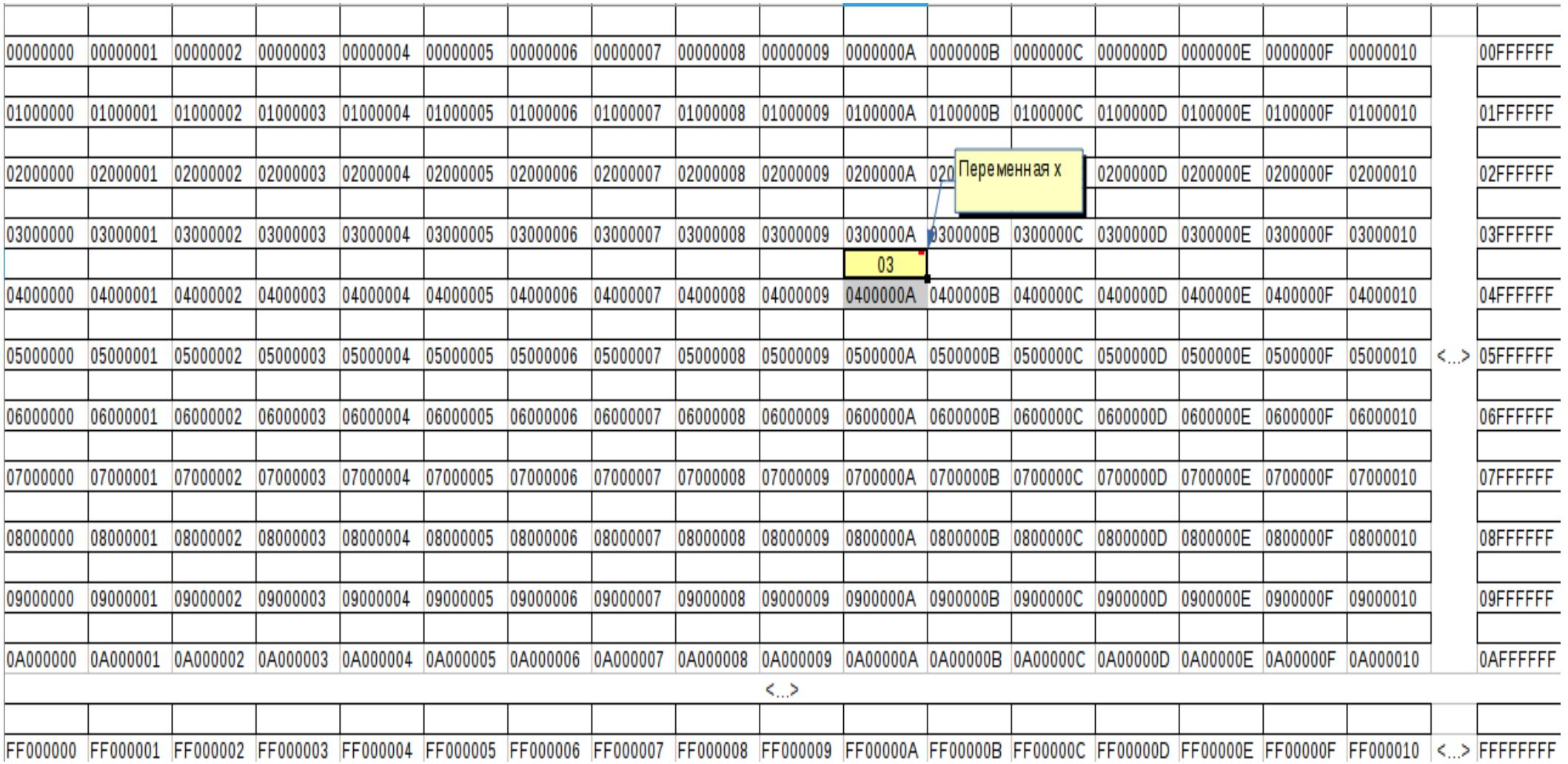
- Тем не менее, следует отметить, что исследование курса мировых информационных ресурсов включает в себя веб-программирование
- Цель курса: овладеть навыками создания сайтов в интернете для представления информационных ресурсов на мировом уровне, а также для обеспечения процесса управления этими ресурсами посредством проектирования, разработки и тестирования веб-приложений
- Без знания основ программирования невозможно грамотно осуществить реализацию конечной структуры сайта или веб-приложения, реализовать общий план взаимодействия всех его компонентов, поскольку программирование является связующим звеном для обеспечения работы компонентов сайта и обеспечения корректного поведения веб-приложения

Основные понятия

- Оперативная память – временное хранилище данных для программ, распределяемое между ними средствами ОС
- Адрес памяти – смещение в оперативной памяти, по которому можно достоверно найти искомые данные, устанавливается средствами ОС
- Переменная – именованная в рамках программы область хранения изменяемых данных, которой соответствует некоторый участок оперативной памяти
- Операция присваивания – задает области памяти, связанной с переменной, определенное значение (обозначим эту операцию как ‘ := ’)

Оперативная память и переменная

- Пример присваивания:
  - x:=3;



Первичные операции

- Введем
  - базовые арифметические операции +, - , * , / и скобки ()
  - операции сравнения =, <, >, <=, >=, <>
  - логические операции AND, OR, NOT и понятия истинности (TRUE) и ложности (FALSE)
  - понятие условия и соответствующий набор операторов IF ... THEN ... ELSE ...
  - понятие метки (LABEL n) и безусловного перехода (GOTO n)
  - ввод (INPUT) и вывод (PRINT) данных
  - комментарий { Произвольный текст }
  - конец программы END
  - ‘;’ в конце строки (ставить можно, но не обязательно)

Таблицы истинности

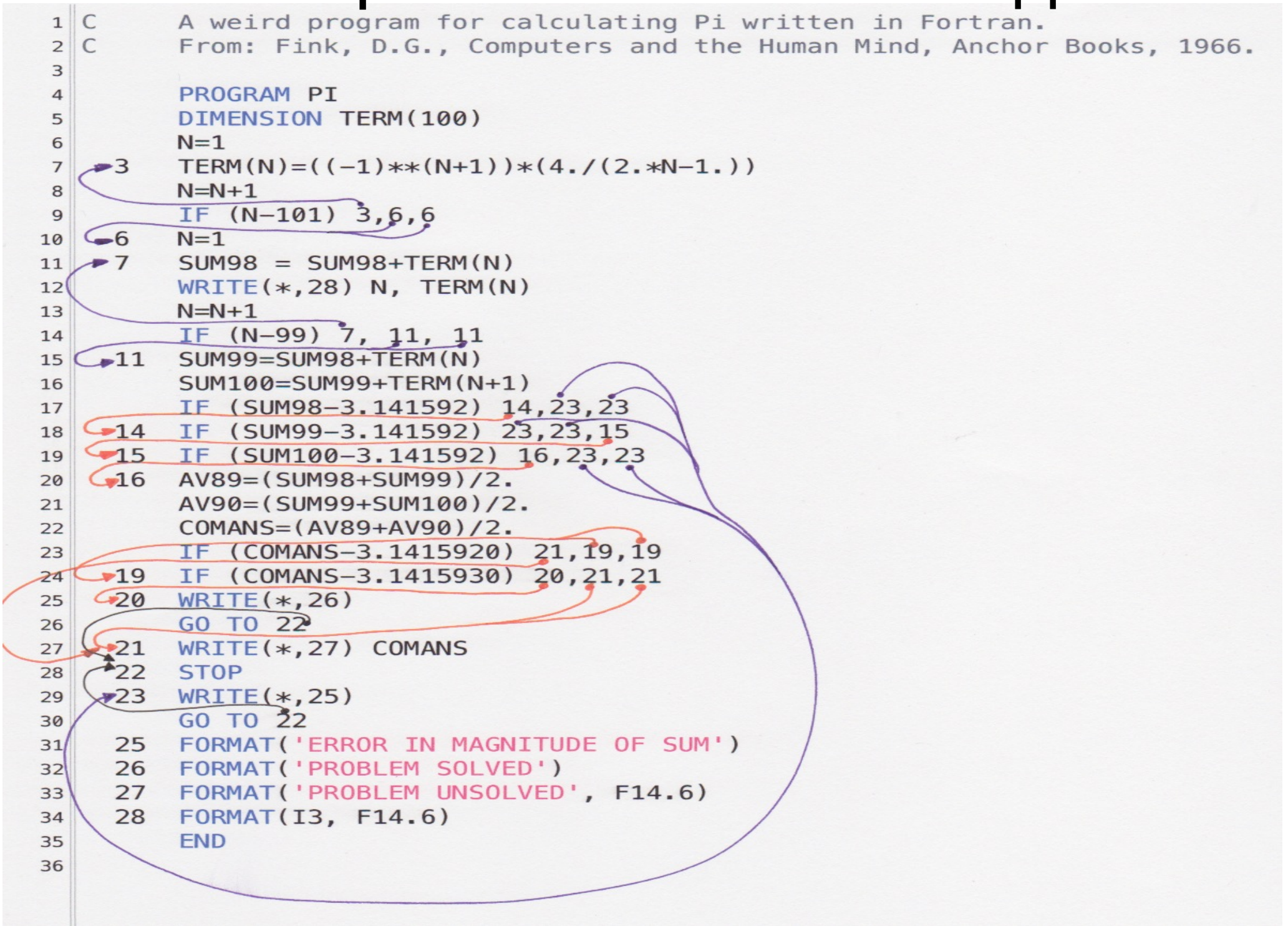
- Для логических функций NOT, AND, OR:

Условие A	***Условие B***	**NOT B**	**A AND B**	**A OR B**
FALSE	*FALSE*	TRUE	FALSE	FALSE
FALSE	*TRUE*	FALSE	FALSE	TRUE
TRUE	*FALSE*	TRUE	FALSE	TRUE
TRUE	*TRUE*	FALSE	TRUE	TRUE

Демонстрация возможностей

- a:=2+2; {a=4 → TRUE}
- IF a>3 THEN PRINT a; {Напечатает a}
- LABEL 10; {Объявим здесь метку с номером 10}
- INPUT b; {Введем значение с клавиатуры в переменную b}
- IF (b>=3) AND (b<=4) THEN PRINT b
  ELSE GOTO 10; {Напечатает b при условии, что b принадлежит [3;4], иначе перейти на строчку с меткой 10 и продолжать выполнение программы}
- END; {Завершаем выполнение программы}

Осторожно: “спагетти-код”



```
C      A weird program for calculating Pi written in Fortran.
C      From: Fink, D.G., Computers and the Human Mind, Anchor Books, 1966.

       PROGRAM PI
       DIMENSION TERM(100)
       N=1
    3  TERM(N)=((-1)**(N+1))*(4./(2.*N-1.))
       N=N+1
       IF (N-101) 3,6,6
    6  N=1
    7  SUM98 = SUM98+TERM(N)
       WRITE(*,28) N, TERM(N)
       N=N+1
       IF (N-99) 7, 11, 11
   11  SUM99=SUM98+TERM(N)
       SUM100=SUM99+TERM(N+1)
       IF (SUM98-3.141592) 14,23,23
   14  IF (SUM99-3.141592) 23,23,15
   15  IF (SUM100-3.141592) 16,23,23
   16  AV89=(SUM98+SUM99)/2.
       AV90=(SUM99+SUM100)/2.
       COMANS=(AV89+AV90)/2.
       IF (COMANS-3.1415920) 21,19,19
   19  IF (COMANS-3.1415930) 20,21,21
   20  WRITE(*,26)
       GO TO 22
   21  WRITE(*,27) COMANS
   22  STOP
   23  WRITE(*,25)
       GO TO 22
   25  FORMAT('ERROR IN MAGNITUDE OF SUM')
   26  FORMAT('PROBLEM SOLVED')
   27  FORMAT('PROBLEM UNSOLVED', F14.6)
   28  FORMAT(I3, F14.6)
       END
```

Внесем изменения в наш язык

- Отказ от использования GOTO и LABEL
- Введение понятия цикла
  - с предусловием (WHILE ... END WHILE)
  - с постусловием (DO ... WHILE)
  - счетчика (FOR … END FOR)
- Введение понятия функции (FUNCTION) и ее частного случая – процедуры (PROCEDURE)
- Введение блочного условия
  IF ... THEN ... ELSE ... END IF
- Введение понятий пропуска итерации цикла (CONTINUE) и выхода из цикла (BREAK)

Примеры структуризации: цикл FOR

```
FOR I:=1 TO 5 STEP 2
    PRINT I
END FOR
{<...>}
```

```
I:=1
LABEL 1
PRINT I
IF I>=5 THEN GOTO 2
I:=I+2
GOTO 1
LABEL 2
{<...>}
```

```
FOR I:=1 TO 5 {STEP 1}
    IF I=2 THEN CONTINUE
    PRINT I {Числа 1 и 3}
    IF I=4 THEN BREAK
END FOR
{<...>}
```

```
I:=1
LABEL 1
IF I=2 THEN GOTO 1
PRINT I
IF I=4 THEN GOTO 2
IF I>=5 THEN GOTO 2
I:=I+1
GOTO 1
LABEL 2
{<...>}
```

Примеры структуризации: цикл WHILE

```
DO                                  |   LABEL 1
    INPUT X                         |   INPUT X
    PRINT X                         |   PRINT X
WHILE X<>0                          |   IF X<>0 THEN GOTO 1
{<...>}                             |   {<...>}
```

```
N:=0                                |   N:=0
X:=0                                |   X:=0
WHILE (N<5) OR (X<>0)               |   LABEL 2
    INPUT X                         |   IF NOT ((N<5) OR (X<>0)) THEN GOTO 1
    PRINT X                         |   INPUT X
    N:=N+1                          |   PRINT X
END WHILE                           |   N:=N+1
{<...>}                             |   GOTO 2
                                    |   LABEL 1
                                    |   {<...>}
```

Примеры структуризации: блочное условие IF ... END IF

```
INPUT A
INPUT B
IF B<>0 THEN
    PRINT A
    PRINT B
    PRINT A/B
ELSE
    PRINT "B=0!"
    PRINT "CAN'T DIVIDE!"
END IF
{<...>}
```

```
INPUT A
INPUT B
IF B<>0 THEN GOTO 1 ELSE GOTO 3
LABEL 2
{<...>}
END
LABEL 1
PRINT A
PRINT B
PRINT A/B
GOTO 2
LABEL 3
PRINT "B=0!"
PRINT "CAN'T DIVIDE!"
GOTO 2
{<...>}
```

Примеры структуризации: функции и процедуры

```
FUNCTION ASKNAME(SIGN)
PRINT "WHAT'S YOUR NAME?"
INPUT N
ASKNAME:=N+SIGN
END FUNCTION

PROCEDURE HELLO
X:=ASKNAME("!")
PRINT "HELLO, "+X
END PROCEDURE

HELLO
PRINT "BYE!"
END
```

```
GOTO 1
LABEL 2
PRINT "WHAT'S YOUR NAME?"
INPUT N
ASKNAME:=N+SIGN {Конкатенация}
GOTO 4
LABEL 3
SIGN:="!"
GOTO 2
LABEL 4
X:=ASKNAME
PRINT "HELLO, "+X
GOTO 5
LABEL 1
GOTO 3
LABEL 5
PRINT "BYE!"
END
```

Зачем все это было нужно

- Псевдоязык позволяет абстрагироваться от заданных языковых конструкций конкретного языка и вникнуть в суть программных решений, перенося их потом на любой изученный язык программирования
- Неструктурированный код встречается в низкоуровневом (близком к аппаратному уровню) программировании из-за ограниченности средств выражения кода; поэтому часто бывает полезно знать, каким структурам может соответствовать тот или иной неструктурированный код
- Объектно-ориентированный подход часто бывает переоценен, а для ряда задач – избыточен
- Для наших текущих целей в рамках данного курса достаточно разбираться в процедурном подходе и основных структурах программирования

Пример изученных конструкций на языке JavaScript

```
<script>
                                                /* Комментарии начинаются со знака // до конца строки
                                                   либо оформляются как этот, многострочный */
function Hello(n){                              // Функция
    return "Hello, "+n+"!";                     // Возвращает строку
}
function Count(lim,block){                      // Блок отделяется фигурными скобками
    var s='';                                   // Переменные объявляются и инициализируются с помощью var
    for(var i=1;i<=lim;++i){                    // Цикл FOR, ++i ~> i=i+1
        if(!(i%2)){continue;}                   // ...или i%2!=0, т.е. пропускаем четные
        if(i==block){break;}
        s+=i+' ';                               // Конкатенируем строку
    }
    return s;
}
a=(1+2-3)*4/5;                                  // a=0
alert(Count(15,a));                             // Вызов функции и вывод результата
for(var i=0;i<2;++i){
    alert((i==0||i>2)+' '+((i>-1)&&(i<1))+';'); // true true; false false;
}
name=prompt("Enter your name:");                // Ввод данных
alert(Hello(name));
do{                                             // Пример цикла с пост-условием
    age=parseInt(prompt("Enter your age:"));
}while(age<0);
</script>
```

Дополнительные сведения об особенностях JavaScript

- Escape-последовательности: \t - табуляция , \n – новая строка, \’ = ‘, \” = “, \\ = \,
- Сдвоенное присваивание *= /= += -= %=
- null – специальное, пустое значение (отсутствие данных)
- undefined – значение не определено (сравнивать через ===)
- сравнение по значению ==, сравнение по значению и типу ===, ! → NOT, && → AND, || → OR, != / !== - работает в сравнениях аналогично сдвоенному присваиванию
- ? - тернарный оператор; a=b?c:d; равносильно if (b) a=c; else a=d; (т.е. условие b проверяется на true/false, соответственно)
- Массив (подвид объекта): var array1 = ["Element1", "Element2"];
- Объект: var object1 = {name1:"Value1", name2:"Value2"; method1: function(){}};
- Свойства объекта отличаются от его методов тем, что методы определяются как функции, а вызов метода осуществляется так же, как и доступ к значению свойства: object1.name1; object1.method1();
- Вызов функции осуществляется через скобки даже при отсутствии входных параметров
- Сравнение двух объектов всегда ложно

Что может делать JavaScript с точки зрения веб-разработки?

- JavaScript – один из “трех китов” веб-разработчика:
  - HTML определяет содержимое веб-страниц
  - CSS контролирует особенности верстки веб-страниц
  - JavaScript задает поведение веб-страниц
- Современный JavaScript на платформе NodeJS способен с высокой скоростью выполнять все необходимые задачи на стороне сервера, а не только браузера-клиента.
- Обработчики событий могут использоваться для верификации и контроля пользовательского ввода и прочих действий
  - которые необходимо сделать при загрузке страницы;
  - которые требуется сделать при закрытии страницы;
  - которые должны быть предприняты при нажатии на некоторую кнопку или другую активную область
- Есть множество способов для обеспечения обработки событий средствами JavaScript:
  - Прямое выполнение через атрибуты событий HTML
  - Вызов функции через атрибуты событий HTML
  - Самостоятельная установка функции-обработчика события для элемента HTML
  - Предотвращение распространения и обработки событий по требованию
- ...и многое другое...

Пример использования JavaScript

- Код страницы:

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<script>
function ButtonPress(){
    var n=document.getElementsByName("Name")[0].value;
    var i=parseInt(parseInt(document.getElementsByName("Age")[0].value)/100); // Could be parseFloat for Float numbers
    var b=document.getElementById("HelloCheck").checked;
    document.getElementById("TextOut").innerHTML=(b?"Hello, "+n+", y":"Y")+"our age in centuries is "+String(i);
}
function CreateHtml(){
    document.write([
    "<body>",
    "<pre>",
    "Name:<input type='text' name='Name' autofocus><br>",
    "Age: <input type='text' name='Age'><br>",
    "<input id='HelloCheck' type='checkbox'>Greet me, please!",
    "<br><button onclick='ButtonPress()'>Click me!</button>",
    "<br><textarea id='TextOut' cols='24' rows='3' disabled></textarea>",
    "</pre>",
    "</body>",
    ].join('\n'));
}
CreateHtml();
</script>
</html>
```

Пример использования JavaScript

- Результат:

Еще один рабочий пример

- Код:

```
<!DOCTYPE html>
<html>
<body>
<p id="answer"></p>
</body>
<script>
var strings=[];
var quantity=Math.random()*100+1;
for(var i=1;i<=quantity;i++){
    var num1=parseInt(Math.random()*9+1);
    var num2=parseInt(Math.random()*9+1);
    strings.push(String(num1)+'*'+String(num2)+'='+String(num1*num2));

}
strings.push('Total: '+String(strings.length)+' lines (without this one)'); // length - возвращает длину массива (или строки)
document.getElementById("answer").innerHTML=strings.join('<br>');
</script>
</html>
```

```
// Math.random() -> [0;1) * 9 ->
//   -> [0;9) +1 -> [1;10)
/* push добавляет элемент в конец массива;
   pop - удаляет */
```

- Результат:

8*6=48
2*1=2
1*6=6
8*9=72
3*6=18
2*5=10
9*8=72
1*3=3
9*2=18
Total: 9 lines (without this one)