

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:
– $x := 3$;

00000000	00000001	00000002	00000003	00000004	00000005	00000006	00000007	00000008	00000009	0000000A	0000000B	0000000C	0000000D	0000000E	0000000F	00000010	00FFFFFF
01000000	01000001	01000002	01000003	01000004	01000005	01000006	01000007	01000008	01000009	0100000A	0100000B	0100000C	0100000D	0100000E	0100000F	01000010	01FFFFFF
02000000	02000001	02000002	02000003	02000004	02000005	02000006	02000007	02000008	02000009	0200000A	0200000B	0200000C	0200000D	0200000E	0200000F	02000010	02FFFFFF
03000000	03000001	03000002	03000003	03000004	03000005	03000006	03000007	03000008	03000009	0300000A	0300000B	0300000C	0300000D	0300000E	0300000F	03000010	03FFFFFF
04000000	04000001	04000002	04000003	04000004	04000005	04000006	04000007	04000008	04000009	0400000A	0400000B	0400000C	0400000D	0400000E	0400000F	04000010	04FFFFFF
05000000	05000001	05000002	05000003	05000004	05000005	05000006	05000007	05000008	05000009	0500000A	0500000B	0500000C	0500000D	0500000E	0500000F	05000010	05FFFFFF
06000000	06000001	06000002	06000003	06000004	06000005	06000006	06000007	06000008	06000009	0600000A	0600000B	0600000C	0600000D	0600000E	0600000F	06000010	06FFFFFF
07000000	07000001	07000002	07000003	07000004	07000005	07000006	07000007	07000008	07000009	0700000A	0700000B	0700000C	0700000D	0700000E	0700000F	07000010	07FFFFFF
08000000	08000001	08000002	08000003	08000004	08000005	08000006	08000007	08000008	08000009	0800000A	0800000B	0800000C	0800000D	0800000E	0800000F	08000010	08FFFFFF
09000000	09000001	09000002	09000003	09000004	09000005	09000006	09000007	09000008	09000009	0900000A	0900000B	0900000C	0900000D	0900000E	0900000F	09000010	09FFFFFF
0A000000	0A000001	0A000002	0A000003	0A000004	0A000005	0A000006	0A000007	0A000008	0A000009	0A00000A	0A00000B	0A00000C	0A00000D	0A00000E	0A00000F	0A000010	0AFFFFFF
<...>																	
FF000000	FF000001	FF000002	FF000003	FF000004	FF000005	FF000006	FF000007	FF000008	FF000009	FF00000A	FF00000B	FF00000C	FF00000D	FF00000E	FF00000F	FF000010	FFFFFFFF

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:

<i>Условие A</i>	<i>Условие B</i>	NOT B	A AND B	A OR B
<i>FALSE</i>	<i>FALSE</i>	TRUE	FALSE	FALSE
<i>FALSE</i>	<i>TRUE</i>	FALSE	FALSE	TRUE
<i>TRUE</i>	<i>FALSE</i>	TRUE	FALSE	TRUE
<i>TRUE</i>	<i>TRUE</i>	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”

```
1 C      A weird program for calculating Pi written in Fortran.
2 C      From: Fink, D.G., Computers and the Human Mind, Anchor Books, 1966.
3
4      PROGRAM PI
5      DIMENSION TERM(100)
6      N=1
7      TERM(N)=((-1)**(N+1))*(4./(2.*N-1.))
8      N=N+1
9      IF (N-101) 3,6,6
10     N=1
11     SUM98 = SUM98+TERM(N)
12     WRITE(*,28) N, TERM(N)
13     N=N+1
14     IF (N-99) 7, 11, 11
15     SUM99=SUM98+TERM(N)
16     SUM100=SUM99+TERM(N+1)
17     IF (SUM98-3.141592) 14,23,23
18     IF (SUM99-3.141592) 23,23,15
19     IF (SUM100-3.141592) 16,23,23
20     AV89=(SUM98+SUM99)/2.
21     AV90=(SUM99+SUM100)/2.
22     COMANS=(AV89+AV90)/2.
23     IF (COMANS-3.1415920) 21,19,19
24     IF (COMANS-3.1415930) 20,21,21
25     WRITE(*,26)
26     GO TO 22
27     WRITE(*,27) COMANS
28     STOP
29     WRITE(*,25)
30     GO TO 22
31     25  FORMAT('ERROR IN MAGNITUDE OF SUM')
32     26  FORMAT('PROBLEM SOLVED')
33     27  FORMAT('PROBLEM UNSOLVED', F14.6)
34     28  FORMAT(I3, F14.6)
35     END
36
```

The diagram illustrates the concept of "spaghetti code" by showing the flow of execution through a Fortran program. The code is annotated with numbers and arrows indicating jumps and loops. A large purple oval encloses the main loop structure, while red arrows highlight specific conditional jumps and loops. The jumps are as follows: Line 9 (IF) jumps to line 3, line 6, or line 6; Line 14 (IF) jumps to line 7, line 11, or line 11; Line 17 (IF) jumps to line 14, line 23, or line 23; Line 18 (IF) jumps to line 23, line 23, or line 15; Line 19 (IF) jumps to line 16, line 23, or line 23; Line 23 (IF) jumps to line 21, line 19, or line 19; Line 24 (IF) jumps to line 20, line 21, or line 21; Line 26 (GO TO) jumps to line 22; Line 30 (GO TO) jumps to line 22. The red arrows highlight the following jumps: Line 17 (IF) to line 14, line 23, or line 23; Line 18 (IF) to line 23, line 23, or line 15; Line 19 (IF) to line 16, line 23, or line 23; Line 23 (IF) to line 21, line 19, or line 19; Line 24 (IF) to line 20, line 21, or line 21; Line 26 (GO TO) to line 22; Line 30 (GO TO) to line 22.

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
    INPUT X
    PRINT X
WHILE X<>0
{<...>}
```

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

```
N:=0
X:=0
WHILE (N<5) OR (X<>0)
    INPUT X
    PRINT X
    N:=N+1
END WHILE
{<...>}
```

```
| N:=0
| X:=0
| LABEL 2
| IF NOT ((N<5) OR (X<>0)) THEN GOTO 1
| INPUT X
| PRINT X
| 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>

```
function Hello(n){
    return "Hello, "+n+"!";
}
function Count(lim,block){
    var s='';
    for(var i=1;i<=lim;++i){
        if(!(i%2)){continue;}
        if(i==block){break;}
        s+=i+' ';
    }
    return s;
}
a=(1+2-3)*4/5;
alert(Count(15,a));
for(var i=0;i<2;++i){
    alert((i==0||i>2)+' '+((i>-1)&&(i<1))+';');
}
name=prompt("Enter your name:");
alert(Hello(name));
do{
    age=parseInt(prompt("Enter your age:"));
}while(age<0);
</script>
```

```
/* Комментарии начинаются со знака // до конца строки
   либо оформляются как этот, многострочный */
// Функция
// Возвращает строку

// Блок отделяется фигурными скобками
// Переменные объявляются и инициализируются с помощью var
// Цикл FOR, ++i ~> i=i+1
// ...или i%2!=0, т.е. пропускаем четные

// Конкатенируем строку

// a=0
// Вызов функции и вывод результата

// true true; false false;

// Ввод данных

// Пример цикла с пост-условием
```

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:

Name:

Age:

☒ Greet me, please!

```
Hello, Serge, your age in  
centuries is 0
```

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)
```