|  |  |  |
| --- | --- | --- |
| **Занятие 1.** Основные понятия языка программирования JavaScript | 01.06; 02.06 | 14.00:14.30 |
| **Занятие 2.** Программирование алгоритмов линейной структуры. Задача «Бронза». | 03.06; 04.06 | 14.00:14.30 |
| **Занятие 3.** Программирование вспомогательных алгоритмов. Задача «Светофор». | 05.06; 08.06 | 14.00:14.30 |
| **Занятие 4.** Программирование разветвляющихся алгоритмов. Задача «Книжный магазин». | 09.06; 10.06 | 14.00:14.30 |
| **Занятие 5.** Программирование алгоритмов циклической структуры. Задача «Факториал». | 11.06; 15.06 | 14.00:14.30 |

**Тайны JavaScript на Web-страницах. Базовые структуры программирования**

**Занятие 1. Основные понятия языка программирования JavaScript**

*Задание:*

*Прочитать и конспектировать темы*

1.Базовые структуры программирования

2.Определение и назначение языка JavaScript

3.Структурные элементы языка

4. Переменные. Выражения и операции

5. Вывод информации методом document.write

6. Ввод информации методом prompt

*Ответить на вопросы, анализировать выводы*

1. Базовые структуры программирования

Алгоритм – чёткое описание последовательности действий для решения поставленной задачи. Умение мыслить *алгоритмически* – представить сложное действие в последовательной совокупности простых действий.

Существуют следующие способы описания алгоритмов:

1. *Словесная инструкция*.
2. *Блок-схема*. Язык блок-схем является универсальным языком описания алгоритмов. Для обозначения действий в блок-схеме приняты следующие элементы:

Ввод данных в переменные.

Вывод результатов и текста.

Простое действие (вычисление и присваивание).

Условие.

Начало (конец) алгоритма.

1. *Программа* – это алгоритм, записанный на каком-либо языке программирования.

Программировани**е** – вид профессиональной деятельности, задача которой – создание новых программ, системных или прикладных.

Система программирования – среда, которая позволяет создавать в памяти компьютера тексты программ, транслировать (переводить) тексты программ в машинные коды, отлаживать и исполнять программы.

Язык программирования – способ записи алгоритма (система обозначений) для выполнения его компьютером.

Языки программирования *высокого уровня*, например, Basic, Fortran, C, Pascal, Java, JavaScript, являются искусственными языками со строго определённым синтаксисом и семантикой. Понятия и структура языков высокого уровня удобны для восприятия человеком.

Любой язык программирования базируется на основных алгоритмических структурах:

1. *Линейность* (следование).
2. *Ветвления* (условные алгоритмы).
3. *Циклы* (повторение).
4. *Подпрограммы (функции).*

Линейные алгоритмы

В линейных алгоритмах действия (команды или операторы) выполняются последовательно, друг за другом: *ввод данных*, их последовательная *обработка* и *вывод результатов* обработки.

Ввод Команда 1 … Команда N Вывод

Условные алгоритмы

Условный алгоритм проверяет истинность заданного условия. Если результат логического выражения в условии равен значению *True*, то выполняется *Блок операторов №1*, иначе (*False*) – *Блок операторов №2.*

*True*  *False*

Условие

Блок операторов №1 Блок операторов №2

Циклические алгоритмы

Циклический алгоритм построен на базе условия. Блок команд, записанный в теле цикла, выполняется заданное число раз.

*False*

Условие

*True*

*Тело цикла:*

Блок команд

Выход из цикла

Вспомогательные алгоритмы - подпрограммы

Во многих языках программирования реализована возможность задания функций-подпрограмм. Подпрограмма – это группа операторов, выполняющих определённые действия, по завершению которых возвращается некоторое значение. Для выполнения подпрограммы необходимо осуществить её вызов.

*Функция-подпрограмма* Средство *вызова функции*

для её выполнения

Тело функции

# Контрольные вопросы

1. Дайте определение алгоритма. Что значит мыслить алгоритмически?
2. Как описывают алгоритмы?
3. Какие блоки приняты для описания алгоритмов блок-схемой?
4. Чем отличается алгоритм от программы?
5. Дайте определение языка программирования.
6. Что такое «система программирования»?
7. Что такое «трансляция» программы? Что такое «отлаживание» программы?
8. Назовите основные типы алгоритмов. Механизмы управления.
9. Приведите собственные примеры алгоритмов.
10. Изобразите алгоритмы в виде блок-схем: «Светофор», «Работа с 8 до 12».

#### Выводы

Вы узнали о базовых понятиях программирования:

* алгоритм – закон последовательности действий для решения задачи
* блок-схема – запись алгоритма на языке блоков с указанием их взаимосвязи
* программа – алгоритм, записанный на языке программирования
* язык программирования – система обозначений для записи алгоритмов
* система программирования – среда компьютера для создания и исполнения программ
* основные типы алгоритмов – линейные, условные, циклические, подпрограммы.

2. Определение и назначение языка JavaScript

Назначение. Язык HTML предоставляет средства для разработки *статических* Web-страниц. Язык программирования JavaScript расширяет возможности HTML для создания *динамических* Web-страниц.

Назначение JavaScript – позволить разработчикам страниц писать небольшие *скрипты* (программы), которые могли бы выполняться не на сервере, а в *среде броузеров*. Такой язык удобен для создания приложений типа клиент–сервер в Internet. В приложениях клиента скрипты встраивают в HTML-документ и используют для обработки событий, связанных с вводом и просмотром информации. Например, приложение, собирающее данные из какой-либо формы, может проверять правильность этих данных до того, как их передавать на сервер. Это улучшает производительность работы Web-документа, так как на сервер передаются только корректные данные.

Определение**.** Язык программирования JavaScript является *объектно-ориентированным* языком, так как основан на использовании объектов.

Программа на языке JavaScript интерпретируется самим броузером Internet Explorer при загрузке документа, в который помещён её код. Интерпретатор JavaScript читает программу сверху вниз и сообщает об ошибках после каждой прочитанной строки. *Интерпретатор* – это переводчик операторов программы в машинные команды.

Возможности. Язык JavaScript позволяет программисту:

* Осуществить обработку информации перед передачей её на сервер.
* Осуществлять диалог с пользователем в среде броузера, без сервера.
* Производить доступ к элементам оформления документа.
* Разгрузить каналы связи.

Встраивание скриптов. Программы на JavaScript встраивают в HTML-документ с помощью пары тегов*:<script>операторы через ;</script>*.

Рекомендуется размещать скрипты в тегах *<head>…</head*>, так как это первый тег, который читает броузер. Однако простые скрипты можно размещать в любом месте тела документа *<body>…</body>*.

3. Структурные элементы языка

Переменные. Переменная – это *поименованная область памяти*, где записываются, хранятся и изменяются значения.

Выражения и операции. *В* *выражении записываются операции над значениями переменных*. Например, в результате арифметических операций, записанных в выражении, в переменную записывается новое значение.

Операторы. *Команды или ключевые слова* для выполнения определённых действий в программе.

Объекты. *Объект*  характеризуется состоянием (набор свойств), поведением (воздействие на другие объекты) и индивидуальностью (различие свойств). Язык JavaScript включает в себя около 20 стандартных объектов. Каждый объект имеет свой арсенал *методов и свойств*. *Методы объекта* выполняют заданные действия над ним. *Свойства объекта* – это совокупность его атрибутов (переменных, значений, других объектов).

События. Событие – это *совершение пользователем каких-либо* *действий* в рамках данной страницы: нажатие и перемещение кнопки мыши, нажатие клавиши. События обрабатываются интерпретатором JavaScript с помощью *обработчиков событий*.

Контрольные вопросы

1. Назначение языка программирования JavaScript.
2. Определение языка программирования JavaScript.
3. Интерпретация скрипта броузером.
4. Каковы возможности языка программирования JavaScript?
5. Как встраиваются скрипты в HTML-документ?
6. Перечислите структурные элементы языка JavaScript.
7. Дайте определения каждой группе структурных элементов языка.

4. Переменные. Выражения и операции

###### Переменные

###### Как и в других языках программирования, в JavaScript переменная имеет два признака:

###### *имя переменной* и *значение переменной*.

###### Имя переменной должно начинаться с латинской буквы или символа подчёркивания \_, следующие символы могут быть латинские буквы или арабские цифры. Язык чувствителен к регистру, то есть следует различать заглавные и маленькие буквы.

*Создать переменную* можно двумя способами:

1. используя ключевое слово *var* *с оператором присваивания* (var a=5; или var a)
2. используя только *оператор присваивания* (а=5)

Оператор присваивания *записывает значение в переменную*.

*Значениями переменных* могут быть:

* *целые и действительные числа*: X1=-5; ax=38; \_S=3.453;
* *арифметические выражения:* X=X\*(X+Y)/2;
* *символьные строки*:

computer=”computer”; z=”язык программирования”; (значения символьных строк выделяются в кавычки)

* *Булевы значения* true и false, а также значение null

При объявлении переменной *не указывается её тип*, язык JavaScript слаботипизирован. Именем переменной не может быть *ключевое слово языка* (названия операторов, объектов и других элементов языка).

##### Выражения и операции

##### Значениями переменных могут быть *арифметические* и *условные* выражения.

*Арифметические выражения* содержат арифметические операции: + сложение (конкатенация для символьных строк), - вычитание, \* умножение, / деление. Приоритет действий выделяют круглые скобки.

Примеры для текстовых переменных:

part1=”Java”; Year=365\*24; x=”ответ равен”+42;

а). part2=”Script” ; б). Minutes=Year\*60; в). z=”58”+42;

part3=part1+part2;

5. Вывод информации методом *document.write*

Язык JavaScript предоставляет способ вывода информации в окно броузера с помощью метода *write* объекта *document*.

document.write(“текст”) – вывод содержимого в кавычках

document.write(z) – вывод значения переменной z

document.write(“текст”+z) – конкатенация переменной с текстом

document.write(“<h1>текст”+z+”</h1>” )– конкатенация переменной с тегами HTML

Используя конкатенацию с тегами форматирования объект *document* предоставляет пользователю доступ к элементам оформления web-страницы.

Вложенные кавычки должны различаться по виду, например:

document.write(“<font color= ‘#red’>текст</font>”);

Для установки цвета фона страницы web-документа используется свойство *bgColor* объекта *document* с указанием кода цвета, например: document.bgColor=”# ffffff”

**Контрольные вопросы**

Что такое переменная? Каковы признаки переменной?

1. Как можно задать имя переменной в языке JavaScript?
2. Как создать переменную? Какой оператор записывает значение в переменную?
3. Какие значения могут содержать переменные?
4. Какие выражения могут содержать переменные? Приведите примеры.
5. Какие арифметические операции выполняют арифметические выражения?
6. Какое средство языка JavaScript позволяет выводить информацию в окно броузера?
7. Какие возможности предоставляет метод *document.write* при выводе информации?
8. Каков синтаксис языка при конкатенации переменных с текстом и тегами HTML?

6. Ввод информации методом *prompt*

**Метод *prompt***

Выводит панель с сообщением и кнопками «Оk» и «Cancel» (отмена). Признак *prompt* – текстовое поле.

Формат в скрипте: b=prompt(“Введите имя”,”Иван”);

Результат в броузере:

Введите имя Ok

Иван Cancel

Первый аргумент в скобках – это сообщение на панели, второй аргумент – это образец ввода текста в текстовом поле. Второй аргумент можно задать пустым, например: b=prompt(“Введите имя”,””);

Выполнение программы продолжается после нажатия одной из кнопок. При этом диалоговая панель исчезает. После обработки функция *prompt* возвращает *текст*, набранный в поле ввода, если была нажата кнопка «Оk» (b=”Иван”), и *null*, если была нажата «Cancel» (b=null)

Метод *prompt* позволяет вводить только текстовые данные, то есть символьную строку. Для работы с числами необходимо применить функцию *parseFloat*, которая преобразует введённую строку в действительное число (или *parseInt* для преобразования в целое число). b= parseFloat(b)или b= parseInt(b)

**Занятие 2.**

**Программирование алгоритмов линейной структуры. Задача «Бронза».**

*Задание:*

*Теория.*

Прочитать и конспектировать темы, ответить на вопросы, анализировать выводы

7. Этапы решения задачи

*Практика.*

Скопируйте выделенный текст программы в html-файле, вставьте в редактор Блокнот (!Поменяйте Wordовские кавычки на Блокнотовские), сохраните с именем Бронза.htm, откройте файл и протестируйте программу несколько раз (нажав в браузере кнопку Обновить) с разными исходными данными.

7. Этапы решения задачи

Решая любую задачу, программист выделяет следующие основные этапы:

1. Описание переменных. Выделение исходных данных, результатов, промежуточных результатов.
2. Построение математической модели решения задачи, последовательности действий.
3. Построение алгоритма и написание программы на языке программирования.
4. Ввод программы в среде программирования.
5. Тестирование программы с различными исходными данными, анализ результатов, отладка программы.

Задача «Бронза»

Для приготовления бронзы надо взять: 17 частей меди, 2 части цинка и 1 часть олова. Сколько надо взять каждого металла для получения К количества бронзы?

I. Описание переменных:

На вводе (исходные данные) – К (количество бронзы);

На выводе (результаты) – М (кол. меди), Z (кол. цинка), О (кол. олова).

Образец вывода в браузере: *Для получения … бронзы возьмите:*

*… меди*

*… цинка*

*… олова*

II. Решение задачи:

1. Какое количество металлов содержится в одной части бронзы? S=K/20;
2. Какое количество меди? M=S\*17;
3. Какое количество цинка? Z=S\*2;
4. Какое количество олова? O=S;

III. Программа в html-файле:

<html>

<head>

</head>

<body>

<script>

k=prompt(“Введите количество бронзы”,”100”);

k=parseInt(k);

s=k/20;

m=s\*17;

z=s\*2;

document.write (“<b>Для получения ”+k+” бронзы возьмите<br>”+m+” меди<br>”+z+” цинка<br>”+s+”олова</b>”)

</script>

</body>

</html>

Контрольные вопросы

1. Определите составные части программы для задачи «Бронза»: ввод данных, обработка данных, вывод результатов.
2. Как выполняются операторы в линейных структурах программирования?
3. Какова функция оператора присваивания?
4. Назовите этапы решения задачи с помощью компьютера.
5. Что такое среда программирования? В какой среде вы создаёте программы на языке JavaScript?

Выводы

* Язык программирования JavaScript расширяет возможности HTML для создания *динамических* Web-страниц, является *объектно-ориентированным* языком. Программы на JavaScript встраивают в HTML-файл с помощью пары тегов*:<script>операторы через ;</script>*. Структурные элементы языка: переменные, выражения и операции, операторы, объекты, события.
* В программах линейной структуры операторы выполняются по порядку их написания. Оператор присваивания является линейным, он записывает значение в переменную. Значением может быть число, текст, арифметическое выражение. Оператор присваивания имеет вид знака =
* Программа линейной структуры должна содержать в себе три части: ввод данных *(prompt*), обработка данных *(вычисление и присваивание*), вывод результатов на экран *(document.write*)
* Любую задачу программист решает в несколько этапов – описание переменных, создание математической модели решения, создание алгоритма-программы, ввод программы в среде программирования, тестирование и отладка программы.

**Занятие 3.**

**Программирование вспомогательных алгоритмов. Задача «Светофор».**

**Задание:**

*Теория.*

Прочитать и конспектировать темы, ответить на вопросы, анализировать выводы

8.Вспомогательный алгоритм. Элемент *function*

*Практика.*

Скопируйте выделенный текст Код для задачи «Имитация светофора», вставьте в редактор Блокнот (!Поменяйте Wordовские кавычки на Блокнотовские), сохраните с именем Светофор.htm, откройте файл и протестируйте программу несколько раз (нажав в браузере кнопку Обновить).

8.Вспомогательный алгоритм. Элемент *function*

Сложные программы размещают в тегах заголовка документа <head>…</head>, используя для этого основной элемент языка – функцию (*function*), которая играет роль *подпрограммы*.

Функция имеет имя, тело функции записывается в фигурных скобках. Для выполнения функции необходим её вызов. Вызов функции можно осуществить с помощью какого-либо обработчика события из тела документа. (Смотри *Схему кода html-документа*)

Таким образом, *функция* считывается в память *во время загрузки* документа в броузер, тот сначала анализирует считанную функцию, а выполняет её *после загрузки* всего документа.

Задача «Имитиция светофора».

Установите цвет фона окна броузера (red, yellow, green) и соответствующее предупреждение (“Стой!”, “Жди!”, “Иди !“) методом alert для обработки сигналов светофора.

Код для задачи «Имитация светофора»

<html>

<head>

<script>

function Start( ) {

document.bgColor=”red”;

alert(“Стой!”);

document.bgColor=”yellow”;

alert(“Жди!”);

document.bgColor=”green”;

alert(“Иди!”)

}

</script>

</head>

<body>

<button onClick=”Start( )”>Светофор</button>

</body>

</html>

Основные способы вызова функции

1.Обработчиком onLoad из тега body <body onLoad=”Start( )”> </body>

2.Дополнительным скриптом из тела документа <body> <script>Start( )</script></body>

3.Обработчиком onClick из тега button (кнопка) <button onClick=”Start( )”>кнопка</button>

4.Функцией javascript из исходного якоря ссылки <a href="javascript: Start( )">ссылка</a>

Выводы

Программы сложных структур выполняются в тегах *<head>…</head>* с помощью функции-подпрограммы *function Имя( ) { тело функции }* . Вызов функции можно осуществить с помощью какого-либо обработчика события, либо дополнительным скриптом из тела документа.

**Занятие 4.**

**Программирование разветвляющихся алгоритмов. Задача «Книжный магазин».**

**Задание:**

*Теория.*

Прочитать и конспектировать темы, ответить на вопросы, анализировать выводы

1.Условный оператор if….

2.Программирование вложенных условий. Задача. «Книжный магазин»

*Практика.*

Скопируйте выделенный текст Код html-файла, вставьте в редактор Блокнот (!Поменяйте Wordовские кавычки на Блокнотовские), сохраните с именем Книжный магазин.htm, откройте файл и протестируйте программу несколько раз (нажав в браузере кнопку Обновить), для разных исходных данных.

1. Условный оператор if….

Часто в задачах требуется выбрать порядок действий в зависимости от условия. Это выполняет *условный оператор*, и такие программы называются программами *разветвляющейся структуры*.

Синтаксис if (условие) *{операторы 1}* else *{операторы 2}*

Механизм управления

1. Условие вырабатывает значение логического выражения: *true* или *false* при проверке *знака сравнения* (**==** равно, **!=** не равно, **>** больше, **<** меньше, **>=** больше или равно, **<=** меньше или равно). Например: сравнение 7>5 вырабатывает значение *true*, а 7<5 вырабатывает значение *false*.
2. Для значения *true* выполняется блок *{операторы 1*}

Для значения *false*  выполняется блок *{операторы 2}*

Блок-схема условного алгоритма

Данные

false условие true

Операторы 2 Операторы 1

Результаты

Неполная форма оператора *if…*

Блок **else** является необязательным, тогда условный оператор имеет вид:

**if (условие) *{операторы}***

Задача “Сравнение чисел”. Даны два числа. Если первое число больше второго, то увеличить эти числа вдвое, в противном случае разделить их на семь.

Описание перменных: a – первое число, b – второе число

Блок-схема:

a, b

false a>b true

a = a/ 7 a = a\*2

b = b/ 7 b = b\*2

a, b

Скрипт:

<html><head><script>

function Start() {

a=prompt(“Введи 1 число”, “”);

a=parseFloat(a);

b=prompt(“Введи 2 число”, “”);

b=parseFloat(b);

document.write(“<h2>Старые числа ”+a+” и ”+b+”</h2>”);

if (a>b) {

a=a\*2;

b=b\*2}

else {

a=a/7;

b=b/7}

document.write(“<h2>Новые числа ”+a+” и ”+b+”</h2>”);

}

</script></head>

<body onLoad=”Start()”>

</body></html>

*Протестируйте программу со следующими исходными данными:*

1. a=10; b=2
2. a=8; b=10

Контрольные вопросы

1. Как выполняется скрипт в тегах заголовка документа *head*?
2. Каков синтаксис элемента *function*?
3. Какое событие происходит, когда страница загружена? Обработчик события?
4. В каких структурах программирования применяется условный оператор?
5. Каков синтаксис оператора *if*?
6. Каков механизм управления условного оператора?
7. Какие операции сравнения выполняет условный оператор?
8. Назовите две формы условного оператора. Изобразите блок-схему неполной формы.

Выводы

Программы сложных структур выполняются в тегах *<head>…</head>* с помощью функции-подпрограммы *function Имя( ) { тело функции }* . В разветвляющихся структурах программирования используется условный оператор *if…*, который проверяет истину операции сравнения в условии. В зависимости от значения логического выражения условия в программе выполняется одна из веток: либо - *true*, либо - *false*.

Программирование вложенных условий. Задача. «Книжный магазин»

Напишите программу, которая работает вместо кассира в книжном магазине. При вводе цены книги и полученной суммы от покупателя программа подсказывает, сколько надо взять сдачи, либо какую сумму покупателю доплатить, либо благодарит за покупку. Протестируйте работу сценария программы в браузере.

Описание переменных. A – цена книги, B – сумма, полученная с покупателя, C – разница между суммой покупателя и ценой книги

Блок-схема

a, b

false true

a или b=null

Запусти

c = b – a программу

false true

c > 0

false true Возьмите

c < 0 сдачу (c)

Благодарю Доплатите

за покупку (с)

Код html-файла

<html><head><script>

function Start() {

var a=prompt('Какова цена книги?',"");

var b=prompt('Сколько вы платите?',"");

if (a==null || b==null) {

alert("Зачем нажал 'Отмена'?\nЗапусти программу ещё раз!");

document.write('<h1 align=center>Хулиган</h1>');

document.bgColor="brown"

} else {

document.write('<b>Книжный магазин<br></b>');

c=parseInt(b)-parseInt(a);

if(c>0) {

document.write("<b>Возьмите сдачу "+c+" у.е.!</b>")

} else {

if (c<0){

document.write("<b>Вы не доплатили "+c+" у.е.!</b>")

} else {

document.write("<b>Благодарим за покупку!</b>")

}

}

document.write('<p><b>Как здорово работать кассиром! </b> </p>');

}

}

</script></head>

<body>

<button onClick=”Start( )”> Книжный магазин </button>

</body></html>

### Занятие 5.

### Программирование алгоритмов циклической структуры. Задача «Факториал».

**Задание:**

*Теория.*

Прочитать и конспектировать темы, ответить на вопросы, анализировать выводы

1. Оператор for…

2. Вычисление сумм и произведений

3.Задача «Факториал».

Скопируйте выделенный текст Код html-файла, вставьте в редактор Блокнот (!Поменяйте Wordовские кавычки на Блокнотовские), сохраните с именем Факториал.htm, откройте файл и протестируйте программу несколько раз (нажав в браузере кнопку Обновить), для разных исходных данных.

### Программирование алгоритмов циклической структуры

1. Оператор for…

*Цикл* – это повторение блока определённых действий заданное число раз.

В языке JavaScript программирование циклов обеспечивают операторы *for*… и *while*…

Синтаксис

for (i=A; i<B; i=i+C) { – заголовок цикла

*блок операторов* – тело цикла

}

где i – *счётчик цикла*, переменная

А – *начальное значение* *счётчика*, число

В – *конечное значение* *счётчика*, число

С – *шаг изменения счётчика*, число.

В з*аголовке цикла* должны быть указаны все параметры цикла. *Тело цикла* записывается в фигурных скобках, здесь перечислены через «;» операторы.

Механизм управления

1. Проверка логического выражения *i<В*  в заголовке цикла
2. Если *i<b* равно *true*, то выполняется *{тело цикла}*, затем происходит изменение (итерация) счётчика *i=i+C* и переход к пункту *1*.

Если *i<b* равно *false*, то управление передаётся *следующему оператору за циклом*.

Пошаговая обработка цикла

Выполнение *тела цикла* для какого-либо значения счётчика *i* называется *шагом цикла*.

Практическая работа «Каковы значения счётчика цикла?»

Как изменится значение счётчика цикла *i* для:

1. А=10 В=25 С=3 при условии i<B
2. А= -2 В= -8 С= -0.5 при условии i>=B
3. А= 1 В=10 С=-2 ( i<B – “зацикливание”, i>B – цикл не выполняется)

Сколько шагов цикла выполнится в каждом случае?

2. Вычисление сумм и произведений

Задача “Вычисление суммы чётных чисел”. Вычислить сумму чётных чисел от *1* до *100*. Распечатать сумму каждого шага цикла.

*Постановка задачи:*  *S = 2+4+6+…+100*

*Скрипт:* <script> function Start( ) {

S=0;

for (i=2; i<=100; i=i+2) {

S=S+i;

document.write (S)

}

} </script>

*Пошаговая обработка цикла:*

На 1 шаге: На 2 шаге: На 3 шаге:

i = 2 2<=100 (true) i = 4 4<=100 (true) i = 6 6<=100 (true)

S=0+2 S=2+4 S=2+4+6

i =2+2 i =4+2 i =6+2

Задача «Среднее арифметическое нечётных чисел». Вычислить среднее арифметическое нечётных чисел от *1* до *N*, где *N* – задано.

*Постановка задачи.* *S=1+3+…+N; K* – количество нечётных чисел; *SA=S/K.*

*Скрипт.* N=prompt (“Введи N”, “”);

N=parseInt(N);

S=0; K=0;

for (i=1; i<=N; i=i+2) {

S=S+i; K=K+1 }

SA=S/K;

document.write (SA) *Выполните первые три шага цикла для N=5.*

3.Задача «Факториал».

Вычислить значение факториала *N!* для заданного числа *N*. Распечатать значение произведения каждого шага цикла по следующему образцу: Произведение … го шага = …

*Постановка задачи*. *P=1\*2\*3\*…\*N*

*Скрипт.*

<html>

<head></head>

<body>

N=prompt (“Введи N”, “”);

N=parseInt(N);

P=1;

for (i=1; i<=N; i++) {

P=P\*i;

document.write (“Произведение”+ i+” го шага =”+ P)

}

</body></html>

### Контрольные вопросы

1. Что называется циклом? Каков синтаксис оператора *for…*? Параметры?
2. Каков механизм управления оператора *for…*? Что называется шагом цикла?
3. Зачем задаются начальные значения переменных при вычислении сумм и произведений?
4. Как вычисляются в цикле суммы и произведения? Что такое *пошаговый вывод* результатов?

**Объекты JavaScript**

|  |  |  |
| --- | --- | --- |
| Занятие 6. Вычисление математических функций. Задача «Функции» | 16.06; 17.06 | 14.00:14.30 |
| Занятие 7. Установка даты и времени на Web-странице. Задача «Часы». | 18.06; 19.06 | 14.00:14.30 |
| Занятие 8. Массивы данных. Типовые задачи. Задача «Телефонный справочник». | 22.06; 23.06 | 14.00:14.30 |
| Занятие 9. Повторение. Тест «Элементы JavaScript» | 25.06; 26.06 | 14.00:14.30 |

Задание

1 Теория: читаем, конспектируем.

2 Практика: создаем html-файл, копируем программный модуль задачи, меняем все кавычки в Блокноте, добавляем необходимые html-теги, добавляем кнопку вызова функции, тестируем программу с разными исходными данными.

**Занятие 6.**

**Вычисление математических функций. Задача «Функции»**

**Объекты**

В языке JavaScript различают две группы объектов: *объекты броузера* (иерархия: window🡪document🡪forms) и *встроенные объекты* (Math, String, Date, Array).Каждый объект имеет свои *свойства и методы*.

**Свойства**

Свойства объектов могут хранить значения (как переменная). Формат: **объект.свойство**

*Пример:* document.bgColor=”red”; , режим – запись; a=document.bgColor;, режим – чтение.

**Методы**

Методы управляют действиями над объектами. Формат: **объект.метод ()**

*Пример:* document.write (“<h1>”+z+”</h1>”);

**Объект *document*** содержит информацию о текущем документе

**Свойства.** *Пример синтасиса:*

1). document.fgColor=”red”;

2). Z= document.lastModified;

* bgColor– цвет фона документа
* fgColor – цвет текста документа
* forms– массив объектов форм документа
* lastModified – дата последнего изменения документа

**Методы.** *Пример синтаксиса:*document.write(z);

* write() – вывод на экран
* clear() – очищает окно, в котором находился документ

**Объект *Math*** предоставляет методы для вычисления математических функций, свойства отсутствуют.

**Методы.** *Пример синтаксиса:* 1). z=Math.min(a,b); 2). document.write(Math.min(a,b));

* abs, sqrt – абсолютное значение, квадратный корень аргумента
* sin, cos, tan – тригонометрические функции (аргументы – в радианах)
* exp, log– экспонента, натуральный логарифм
* min, max – большее или меньшее двух аргументов
* round – округляет аргумент до ближайшего целого *(5,2 → 5; 5,8 → 6)*
* floor– возвращает целое число аргумента, отбрасывая его дробную часть*(5,2 → 5; 5,8 → 5)*

<script>

function matem() {

document.write("<h3><font color=red><center>Математические функции объекта Math</center></font></h3>");

document.fgColor="red";

document.write("<h4><font color=blue>Математические функции (один аргумент)</font></h4>");

N=prompt("Введите число N и узнаете значение некоторых математических функций","");

N=parseFloat(N);

M=Math.sin(N); M=Math.round(M\*100)/100;

L=Math.tan(N); L=Math.round(L\*100)/100;

F=Math.exp(N); F=Math.round(F\*100)/100;

R=Math.sqrt(N); R=Math.round(R\*100)/100

document.write("<table border=2 width=40% bgcolor=lightcyan>");

document.write("<tr><th>Функция</th><th>Результат</th></tr>");

document.write("<tr><td>Sin("+N+")</td><td>"+M+"</td></tr>");

document.write("<tr><td>Tan("+N+")</td><td>"+L+"</td></tr>");

document.write("<tr><td>Exp("+N+")</td><td>"+F+"</td></tr>");

document.write("<tr><td>Sqrt("+N+")</td><td>"+R+"</td></tr>");

document.write("</table>");

alert("Сейчас вы увидите функции, работающие с дробной частью числа");

b=prompt("Введите дробное число",""); b=parseFloat(b);

document.write("<h4><font color=blue>Математические функции (дробный аргумент)</font></h4>");

f1=Math.round(b);

f2=Math.ceil(b);

f3=Math.floor(b);

document.write("<table border=2 width=40% bgcolor=lightcyan>");

document.write("<tr><th>Функция</th><th>Результат</th></tr>");

document.write("<tr><td>round("+b+")</td><td>"+f1+"</td></tr>");

document.write("<tr><td>ceil("+b+")</td><td>"+f2+"</td></tr>");

document.write("<tr><td>floor("+b+")</td><td>"+f3+"</td></tr>");

document.write("</table>");

document.write("Последняя модификация была:"+document.lastModified);

}

</script>

**Занятие 7.**

**Установка даты и времени на Web-странице. Задача «Часы».**

**Объект *Date*** предоставляет методы для установки текущей даты, свойства отсутствуют.

**Методы.** *Пример синтаксиса:* 1). var z=new Date(); v=z.getYear(); 2). document.write(z.toLocaleString());

|  |  |
| --- | --- |
| getYear() – год (число) | getHours() – часы (0-23) |
| getMonth() – месяц (0-11) | getMinutes() – минуты (0-59) |
| getDate() - день (1-31) | getSeconds() – секунды (0-59) |
| toLocaleString() - дата в формате «*22 сентября 2006 г. 11:37:52»* | |

<html><head>

<script>

function Timer() {

var z = new Date();

document.R.data1.value=z.toLocaleString();

v1 = z.getYear();

v2 = z.getMonth()+1;

v3 = z.getDate();

v4 = z.getHours();

v5 = z.getMinutes();

v6 = z.getSeconds();

v = "год:"+v1 + "; месяц:"+ v2 + "; день:"+ v3 + "; час:"+ v4 + "; минута:"+ v5 + "; секунда:"+ v6;

document.R.data2.value=v; document.R.data3.value=z.getHours()+":"+z.getMinutes()+":"+z.getSeconds();

setTimeout('Timer()',1000)

}

</script>

</head>

<body onLoad=’Timer()’>

<h1> Объект Date</h1>

<h3> Задача "Часы" </h3>

<h5> Получите на экране интерактивное время.</h5>

<FORM name="R">

Полный формат даты с помощью метода <i>toLocaleString()</i>:<BR>

<INPUT type="text" name="data1" size=25><BR>

Полный формат даты методами объекта Date:<BR>

<INPUT type="text" name="data2" size=50><BR>

Ещё один формат:<BR>

<INPUT type="text" name="data3" size=8 style="background:lime; color:white; font-size:90pt; font-family:Verdana; cursor:default"><BR>

</form>

</body></html>

**Занятие 8.**

**Массивы данных. Типовые задачи. Задача «Телефонный справочник».**

**Объект *Array*** позволяет выполнять обработку массивов данных (последовательностей) А0, А1,… , Аn

Для *создания* массива данных А0, А1,… , Аn применяется объект Arrayи конструктор new

var A=new Array ();

**Свойство:** A.length – длина, количество элементов массива. Нумерация элементов массива начинается с 0.

Для *заполнения* массива с клавиатуры применяют два способа

*2. Присваивание элементам массива их значений*

var A = new Array (5,-1,7);

Выполнение программы приводит к:

A[0]=5; A[1]=-1; A[2]=7;

b= A.length → (в результате b=3)

*1. Ввод значений с клавиатуры*

var A = new Array ();

for (i=0; i<=20; i++) {

A[i]=prompt(“Введи”+i+”-й элемент”,””) }

b= A.length → (в результате b=21)

**Т и п о в ы е з а д а ч и о б р а б о т к и м а с с и в о в**

**I. Поиск суммы элементов массива**

Найти сумму элементов массива a0, a1, …, an-1. Элементы и количество элементов массива ввести с клавиатуры.

n=prompt(“Количество элементов массива?”,””);

n=parseInt (n); s=0; var a = new Array ();

for (i=0; i<=n-1; i++) {

b=prompt (“Введи”+i+”-й элемент массива”,””);

a[i]=parseFloat (b); s=s+a[i] }

document.write (“сумма элементов массива = ”+s);

Алгоритм решения

1. Ввод n и создание объекта Array для переменной a.

2. Организация цикла для ввода и суммирования элементов массива *s=a0+a1+…+an-1*

3. Вывод результата из переменной *s*

**II. Поиск интересующего элемента в массиве**

В массиве из десяти чисел (2,-10,5,5,2,2,5,5,-10,-10) определить номера элементов, равных заданному числу m. Вывести сообщение «Таких данных в массиве нет», если числа m в массиве не окажется.

Алгоритм решения

1. Ввод с клавиатуры целого числа m

2. Проверка условия (m==a[i] ) в цикле и вывод номера элемента.

3. Объявление переменной k с несуществующим номером элемента массива k=–1.

4. Переприсваивание k в цикле при нахождении элемента m.

5. Проверка условия на первоначальное значение k после выполнения цикла.

var a = new Array (2,-10,5,5,2,2,5,5,-10,-10);

m=prompt (“Введите интересующее вас число”,””);

m=parseInt (m);

var k=-1;

for (i=0; i<=a.length -1; i++) **{**

if (m==a[i]) { document.write (i+“-№ числа”+m+”в массиве<br>”);

k=i; }  **}**

if (k==-1) {document.write(“числа”+m+” в массиве нет”)}

Задача «Телефонный справочник».

function z2() {

alert("Поиск телефона по заданной фамилии");

document.write ("<h2><font color=red>Телефонный справочник</font></h2>");

var a = new Array ("Катя","Петя","Вася","Таня","Ваня");

var b = new Array ("021","022","023","024","025");

m=prompt ("Введите интересующее вас имя /Катя,Петя,Вася,Таня,Ваня/","");

var k=-1;

for (i=0; i<=a.length -1; i++) {

if ( m==a[i] ) {document.write ("<br>"+b[i]+"-номер абонента `"+m+"`");

k=i; break }

}

if ( k==-1) { document.write ("Абонент не найден!"); }

}