# Лабораторная №1

## Выведите alert

Сделайте страницу, которая выводит «Я – JavaScript!».

Создайте её на диске, откройте в браузере, убедитесь, что всё работает.

## Вывести alert внешним скриптом

Возьмите решение предыдущей задачи

Выведите alert и вынесите скрипт во внешний файл alert.js, который расположите в той же директории.

Откройте страницу и проверьте, что вывод сообщения всё ещё работает.

## Работа с переменными

Объявите две переменные: admin и name.

Запишите в name строку "Василий".

Скопируйте значение из name в admin.

Выведите admin (должно вывести «Василий»).

## Объявление переменных

Создайте переменную для названия нашей планеты и присвойте ей значение "Земля". *Правильное* имя выберите сами.

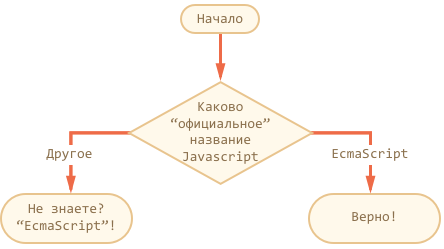
Создайте переменную для имени посетителя со значением "Петя". Имя также на ваш вкус.

## Проверка стандарта

Используя конструкцию if..else, напишите код, который будет спрашивать: «Каково «официальное» название JavaScript?».

Если посетитель вводит «ECMAScript», то выводить «Верно!», если что-то другое – выводить «Не знаете? «ECMAScript»!».

Блок-схема:



## Получить знак числа

Используя конструкцию if..else, напишите код, который получает значение prompt, а затем выводит alert:

1, если значение больше нуля,

-1, если значение меньше нуля,

0, если значение равно нулю.

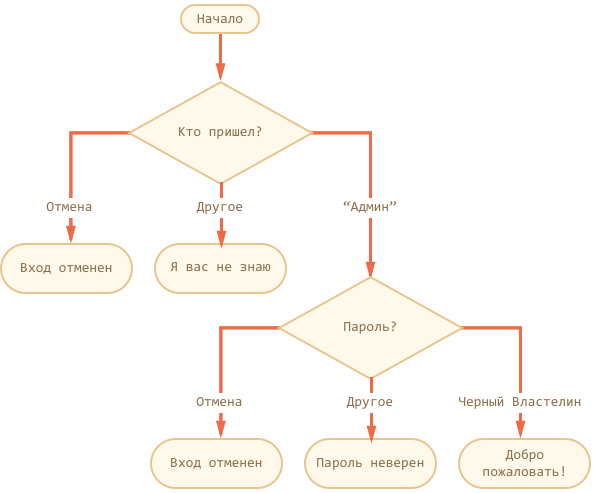
## Проверка логина

Напишите код, который будет спрашивать логин (prompt).

Если посетитель вводит «Админ», то спрашивать пароль, если нажал отмена (escape) – выводить «Вход отменён», если вводит что-то другое – «Я вас не знаю».

Пароль проверять так. Если введён пароль «Чёрный Властелин», то выводить «Добро пожаловать!», иначе – «Пароль неверен», при отмене – «Вход отменён».

Блок-схема:



Для решения используйте вложенные блоки if. Обращайте внимание на стиль и читаемость кода.

## Перепишите 'if' в '?'

Перепишите if с использованием оператора '?':

if (a + b < 4) {

result = 'Мало';

} else {

result = 'Много';

}

## Перепишите 'if..else' в '?'

Перепишите if..else с использованием нескольких операторов '?'.

Для читаемости – оформляйте код в несколько строк.

var message;

if (login == 'Вася') {

message = 'Привет';

} else if (login == 'Директор') {

message = 'Здравствуйте';

} else if (login == '') {

message = 'Нет логина';

} else {

message = '';

}

## Проверка if внутри диапазона

Напишите условие if для проверки того факта, что переменная age находится между 14 и 90включительно.

«Включительно» означает, что концы промежутка включены, то есть age может быть равна 14 или 90.

## Проверка if вне диапазона

Напишите условие if для проверки того факта, что age НЕ находится между 14 и 90 включительно.

Сделайте два варианта условия: первый с использованием оператора НЕ !, второй – без этого оператора.

## Замените for на while

Перепишите код, заменив цикл for на while, без изменения поведения цикла.

for (var i = 0; i < 3; i++) {

alert( "номер " + i + "!" );

}

## Повторять цикл, пока ввод неверен

Напишите цикл, который предлагает prompt ввести число, большее 100. Если посетитель ввёл другое число – попросить ввести ещё раз, и так далее.

Цикл должен спрашивать число пока либо посетитель не введёт число, большее 100, либо не нажмёт кнопку Cancel (ESC).

Предполагается, что посетитель вводит только числа. Предусматривать обработку нечисловых строк в этой задаче необязательно.

## Вывести простые числа

Натуральное число, большее 1, называется простым, если оно ни на что не делится, кроме себя и 1.

Другими словами, n>1 – простое, если при делении на любое число от 2 до n-1 есть остаток.

Создайте код, который выводит все простые числа из интервала от 2 до 10.

 Результат должен быть: 2,3,5,7.

P.S. Код также должен легко модифицироваться для любых других интервалов.

## Напишите "if", аналогичный "switch"

Напишите if..else, соответствующий следующему switch:

switch (browser) {

case 'IE':

alert( 'О, да у вас IE!' );

break;

case 'Chrome':

case 'Firefox':

case 'Safari':

case 'Opera':

alert( 'Да, и эти браузеры мы поддерживаем' );

break;

default:

alert( 'Мы надеемся, что и в вашем браузере все ок!' );

}

## Переписать if'ы в switch

Перепишите код с использованием одной конструкции switch:

var a = +prompt('a?', '');

if (a == 0) {

alert( 0 );

}

if (a == 1) {

alert( 1 );

}

if (a == 2 || a == 3) {

alert( '2,3' );

}

# Лабораторная №2

## Перепишите функцию, используя оператор '?' или '||'

Следующая функция возвращает true, если параметр age больше 18. В ином случае она задаёт вопрос confirm и возвращает его результат.

function checkAge(age) {

if (age > 18) {

return true;

} else {

return confirm('Родители разрешили?');

}

}

Перепишите функцию, чтобы она делала то же самое, но без if, в одну строку. Сделайте два варианта функции checkAge:

Используя оператор '?'

Используя оператор ||

## Функция min

Задача «Hello World» для функций :)

Напишите функцию min(a,b), которая возвращает меньшее из чисел a,b.

Пример вызовов:

min(2, 5) == 2

min(3, -1) == -1

min(1, 1) == 1

## Функция pow(x,n)

Напишите функцию pow(x,n), которая возвращает x в степени n. Иначе говоря, умножает x на себя n раз и возвращает результат.

pow(3, 2) = 3 \* 3 = 9

pow(3, 3) = 3 \* 3 \* 3 = 27

pow(1, 100) = 1 \* 1 \* ...\*1 = 1

Создайте страницу, которая запрашивает x и n, а затем выводит результат pow(x,n).

P.S. В этой задаче функция обязана поддерживать только натуральные значения n, т.е. целые от 1 и выше.

## Вычислить сумму чисел до данного

Напишите функцию sumTo(n), которая для данного n вычисляет сумму чисел от 1 до n, например:

sumTo(1) = 1

sumTo(2) = 2 + 1 = 3

sumTo(3) = 3 + 2 + 1 = 6

sumTo(4) = 4 + 3 + 2 + 1 = 10

...

sumTo(100) = 100 + 99 + ... + 2 + 1 = 5050

Сделайте три варианта решения:

1. С использованием цикла.
2. Через рекурсию, т.к. sumTo(n) = n + sumTo(n-1) для n > 1.
3. С использованием формулы для суммы арифметической прогрессии.

Пример работы вашей функции:

function sumTo(n) { /\*... ваш код ... \*/ }

alert( sumTo(100) ); // 5050

Какой вариант решения самый быстрый? Самый медленный? Почему?

Можно ли при помощи рекурсии посчитать sumTo(100000)? Если нет, то почему?

## Вычислить факториал

Факториа́л числа – это число, умноженное на «себя минус один», затем на «себя минус два» и так далее, до единицы. Обозначается n!

Определение факториала можно записать как:

n! = n \* (n - 1) \* (n - 2) \* ...\*1

Примеры значений для разных n:

1! = 1

2! = 2 \* 1 = 2

3! = 3 \* 2 \* 1 = 6

4! = 4 \* 3 \* 2 \* 1 = 24

5! = 5 \* 4 \* 3 \* 2 \* 1 = 120

Задача – написать функцию factorial(n), которая возвращает факториал числа n!, используя рекурсивный вызов.

alert( factorial(5) ); // 120

Подсказка: обратите внимание, что n! можно записать как n \* (n-1)!. Например: 3! = 3\*2! = 3\*2\*1! = 6

## Числа Фибоначчи

Последовательность чисел Фибоначчи имеет формулу Fn = Fn-1 + Fn-2. То есть, следующее число получается как сумма двух предыдущих.

Первые два числа равны 1, затем 2(1+1), затем 3(1+2), 5(2+3) и так далее: 1, 1, 2, 3, 5, 8, 13, 21....

Числа Фибоначчи тесно связаны с золотым сечением и множеством природных явлений вокруг нас.

Напишите функцию fib(n), которая возвращает n-е число Фибоначчи. Пример работы:

function fib(n) { /\* ваш код \*/ }

alert( fib(3) ); // 2

alert( fib(7) ); // 13

alert( fib(77)); // 5527939700884757

Все запуски функций из примера выше должны срабатывать быстро.

# Лабораторная №3

## Создайте калькулятор

Создайте объект calculator с тремя методами:

* read() запрашивает prompt два значения и сохраняет их как свойства объекта
* sum() возвращает сумму этих двух значений
* mul() возвращает произведение этих двух значений

var calculator = {

...ваш код...

}

calculator.read();

alert( calculator.sum() );

alert( calculator.mul() );

### 

## Цепочка вызовов

Есть объект «лестница» ladder:

var ladder = {

step: 0,

up: function() { // вверх по лестнице

this.step++;

},

down: function() { // вниз по лестнице

this.step--;

},

showStep: function() { // вывести текущую ступеньку

alert( this.step );

}

};

Сейчас, если нужно последовательно вызвать несколько методов объекта, это можно сделать так:

ladder.up();

ladder.up();

ladder.down();

ladder.showStep(); // 1

Модифицируйте код методов объекта, чтобы вызовы можно было делать цепочкой, вот так:

ladder.up().up().down().up().down().showStep(); // 1

Как видно, такая запись содержит «меньше букв» и может быть более наглядной.

Такой подход называется «чейнинг» (chaining) и используется, например, во фреймворке jQuery.

## Сумма произвольного количества скобок

Напишите функцию sum, которая будет работать так:

sum(1)(2) == 3; // 1 + 2

sum(1)(2)(3) == 6; // 1 + 2 + 3

sum(5)(-1)(2) == 6

sum(6)(-1)(-2)(-3) == 0

sum(0)(1)(2)(3)(4)(5) == 15

Количество скобок может быть любым.

## Создать Calculator при помощи конструктора

Напишите функцию-конструктор Calculator, которая создает объект с тремя методами:

Метод read() запрашивает два значения при помощи prompt и запоминает их в свойствах объекта.

Метод sum() возвращает сумму запомненных свойств.

Метод mul() возвращает произведение запомненных свойств.

Пример использования:

var calculator = new Calculator();

calculator.read();

alert( "Сумма=" + calculator.sum() );

alert( "Произведение=" + calculator.mul() );

## Создайте калькулятор

Напишите конструктор Calculator, который создаёт расширяемые объекты-калькуляторы.

Эта задача состоит из двух частей, которые можно решать одна за другой.

Первый шаг задачи: вызов calculate(str) принимает строку, например «1 + 2», с жёстко заданным форматом «ЧИСЛО операция ЧИСЛО» (по одному пробелу вокруг операции), и возвращает результат. Понимает плюс + и минус -.

Пример использования:

var calc = new Calculator;

alert( calc.calculate("3 + 7") ); // 10

Второй шаг – добавить калькулятору метод addMethod(name, func), который учит калькулятор новой операции. Он получает имя операции name и функцию от двух аргументов func(a,b), которая должна её реализовывать.

Например, добавим операции умножить \*, поделить / и возвести в степень \*\*:

var powerCalc = new Calculator;

powerCalc.addMethod("\*", function(a, b) {

return a \* b;

});

powerCalc.addMethod("/", function(a, b) {

return a / b;

});

powerCalc.addMethod("\*\*", function(a, b) {

return Math.pow(a, b);

});

var result = powerCalc.calculate("2 \*\* 3");

alert( result ); // 8

Поддержка скобок и сложных математических выражений в этой задаче не требуется.

Числа и операции могут состоять из нескольких символов. Между ними ровно один пробел.

Предусмотрите обработку ошибок. Какая она должна быть – решите сами.

## Решето Эратосфена

Целое число, большее 1, называется простым, если оно не делится нацело ни на какое другое, кроме себя и 1.

Древний алгоритм «Решето Эратосфена» для поиска всех простых чисел до n выглядит так:

1. Создать список последовательных чисел от 2 до n: 2, 3, 4, ..., n.
2. Пусть p=2, это первое простое число.
3. Зачеркнуть все последующие числа в списке с разницей в p, т.е. 2\*p, 3\*p, 4\*p и т.д. В случае p=2это будут 4,6,8....
4. Поменять значение p на первое не зачеркнутое число после p.
5. Повторить шаги 3-4 пока p2 < n.
6. Все оставшиеся не зачеркнутыми числа – простые.

Реализуйте «Решето Эратосфена» в JavaScript, используя массив.

Найдите все простые числа до 100 и выведите их сумму.