

МИНОБРНАУКИ РОССИИ

Федеральное государственное образовательное учреждение высшего профессионального образования

**«Московский технологический университет»**

Институт Информационных Технологий

Кафедра «ИНэУМ»

**ОТЧЕТ**

по лабораторной работе

по дисциплине

«Проектирование сред программирования и их компонентов»

|  |  |  |  |
| --- | --- | --- | --- |
| Выполнил студент группы ИКМО-04-2017 | |  | Борзунов Н.С. |
| Принял | к.т.н., доцент | | Данилкин Ф.А. |
| Выполнено | «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2018 | г. |  |
| Зачтено | «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2018 | г. |  |

МОСКВА 2018

**Оглавление**

[Задания лабораторной работы 3](#_Toc515709886)

[Задание №1 3](#_Toc515709887)

[Задание №2 4](#_Toc515709888)

[Задание №3 6](#_Toc515709889)

[Задание №4 8](#_Toc515709890)

[Задание №5 10](#_Toc515709891)

[Задание №6 11](#_Toc515709892)

[Заключение 13](#_Toc515709893)

[Список использованной литературы 13](#_Toc515709894)

**Задания лабораторной работы**

**Задание №1**

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 ||

**Код программы**

Листинг 1.1 - Файл 1.html

<html>

<body>

<script>

function checkAge(age)

{

return age>18 ? true : confirm('Родители разрешили?');

}

function checkAge(age)

{

return (age > 18) || confirm('Родители разрешили?');

}

var age = prompt("Введите age", '');

checkAge(age);

</script>

</body>

</html>

**Демонстрация выполнения программы**

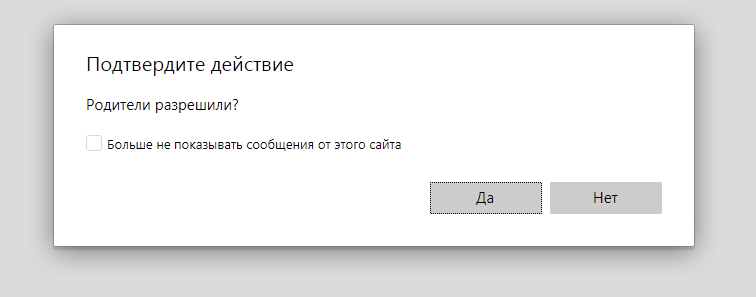


Рисунок 1.1 - Результат выполнения программы

**Задание №2**

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

**Код программы**

Листинг 3.1 - Файл 3.html

<html>

<body>

<script>

var a = prompt("Введите a", '');

var b = prompt("Введите b", '');

function min(a, b)

{

if (a < b)

{

return a;

}

else if (a > b)

{

return b;

}

else

{

return a;

}

}

alert(min(a,b));

</script>

</body>

</html>

**Демонстрация выполнения программы**

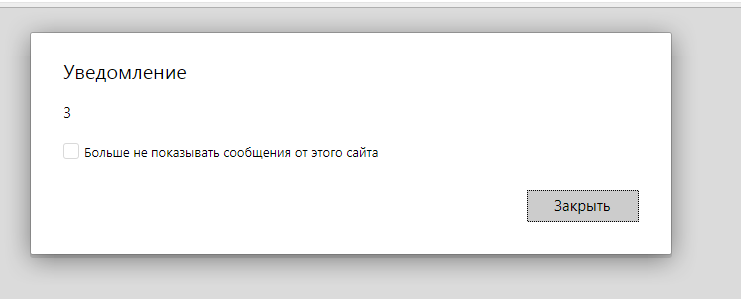


Рисунок 2.1 - Результат выполнения программы

**Задание №3**

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.

**Код программы**

Листинг 3.1 - Файл 3.html

<html>

<body>

<script>

function pow(x, n)

{

var nRes = x;

for (var i = 1; i < n; i++)

{

nRes \*= x;

}

return nRes;

}

var x = prompt("Введите x", '');

var n = prompt("Введите n", '');

while (n<1 && n!= null)

{

n = prompt("Введите натуральное число n", '');

}

alert(pow(x, n));

</script>

</body>

</html>

**Демонстрация выполнения программы**

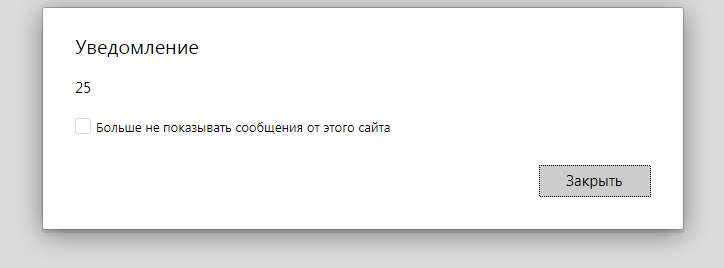


Рисунок 3.1 - Результат выполнения программы

**Задание №4**

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

**Код программы**

Листинг 4.1 - Файл 4.html

<html>

<body>

<script>

function sumTo1(nVal)

{

var sum = 0;

for (var i=1;i<=nVal;i++)

{

sum+=i;

}

return sum;

}

function sumTo2(nVal)

{

if (nVal==0)

{

return 0;

}

else

{

return nVal+sumTo2(nVal-1);

}

}

function sumTo3(nVal)

{

return nVal \* (nVal + 1) / 2;

}

alert(sumTo1(7));

alert(sumTo2(7));

alert(sumTo3(7));

</script>

</body>

</html>

**Демонстрация выполнения программы**

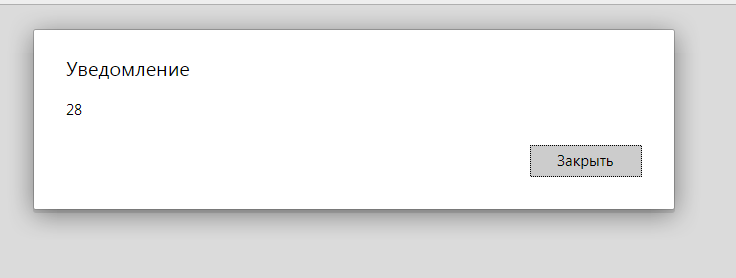


Рисунок 4.1 - Результат выполнения программы

**Задание №5**

Факториал 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

**Код программы**

Листинг 5.1 - Файл 5.html

<html>

<body>

<script>

function factorial(nVal)

{

if (nVal == 1)

{

return 1;

}

return nVal\*factorial(nVal-1);

}

n = prompt("Введите число", '');

alert(factorial(n));

</script>

</body>

</html>

**Демонстрация выполнения программы**

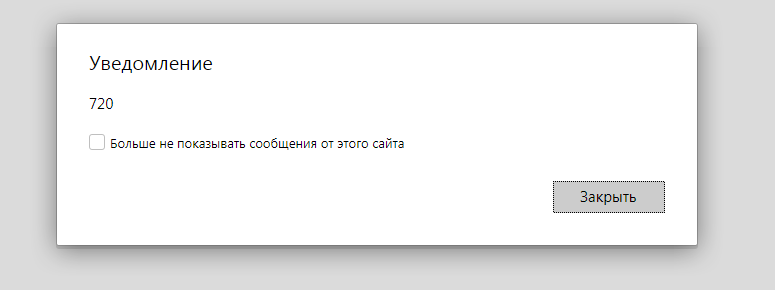


Рисунок 5.1 - Результат выполнения программы

**Задание №6**

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.

**Код программы**

Листинг 6.1 - Файл 6.html

<html>

<body>

<script>

function fib(n)

{

var nVal1 = 1, nRes = 1;

for (var i = 3; i <= n; i++)

{

var nVal2 = nVal1 + nRes;

nVal1 = nRes;

nRes = nVal2;

}

return nRes;

}

n = prompt("Введите число", '');

alert(fib(n));

</script>

</body>

</html>

**Демонстрация выполнения программы**

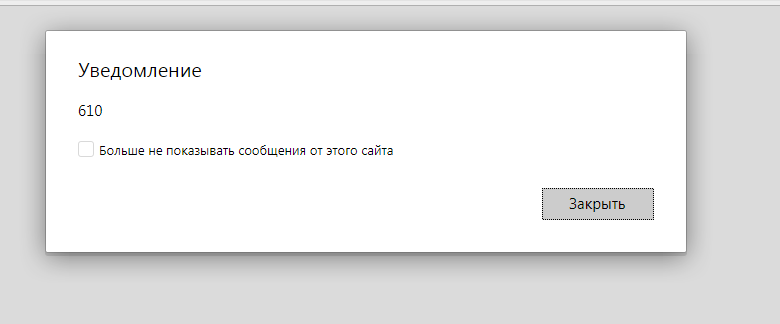


Рисунок 6.1 - Результат выполнения программы

**Заключение**

При выполнении данной лабораторной работы были реализованы задачи переписать функцию, используя оператор '?' или '||', разработать функцию min, функцию pow(x,n), вычислить сумму чисел до данного, вычислить факториал, вычислить число Фибоначчи.

Для задания № 4 решение по формуле работает быстрее всех. Это очевидно. Оно использует всего три операции для любого n, а цикл и рекурсия требуют как минимум n операций сложения.

Вариант с циклом – второй по скорости. Он быстрее рекурсии, поскольку операций столько же, однако отсутствуют дополнительные вычислительные затраты на организацию вложенных вызовов.

Рекурсия в данном случае работает медленнее всех.

Следует отметить, что существует ограничение глубины вложенных вызовов. Таким образом, рекурсивный вызов sumTo(100000) выдаст ошибку.

**Список использованной литературы**

1. Дж. Роббинс HTML5, CSS3 и JavaScript. Исчерпывающее руководство (+ DVD), 2017

2. Э. Браун Изучаем JavaScript. Руководство по созданию современных веб-сайтов, 2017

3. М. Дэвид JavaScript и jQuery. Исчерпывающее руководство. 3-е издание, 2017