

Experiment No. 1

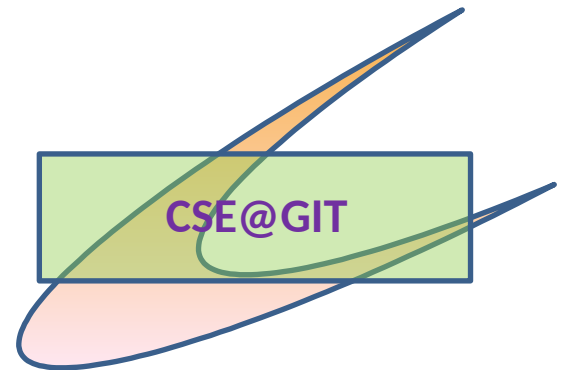
Problem Definition: 1. Develop and demonstrate a XHTML file that includes Javascript script for the following problems:

a) Input: A number n obtained using prompt

Output: The first n Fibonacci numbers

b) Input: A number n obtained using prompt

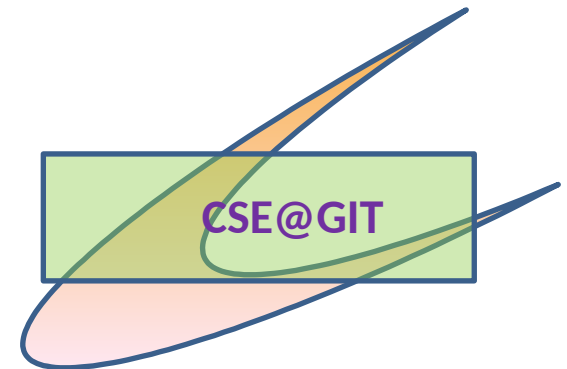
Output: A table of numbers from 1 to n and their squares using
alert



Objectives of the Experiment:

To demonstrate the use of XHTML and JavaScript

To develop an understanding of JavaScript syntax



Need of the Experiment

- JavaScript allows you to build interactive websites
- JavaScript has become an essential web technology along with HTML and CSS, as most browsers implement JavaScript
- For web development, as a front-end developer or on using JavaScript for backend development
- JavaScript usage has now extended to mobile app development, desktop app development, and game development

Theoretical Background of the Experiment

JavaScript

- JavaScript is an interpreted, scripting language, designed to add interactivity to HTML pages
- It is used in Web pages to improve the design, validate forms, detect browsers, create cookies
- Most popular scripting language on the internet, and works on all major browsers like Firefox, Chrome, Opera, IE
- It is usually embedded directly into HTML pages
- Can be used without purchasing a license

Theoretical Background of the Experiment

JavaScript

- JavaScript created by Netscape
- Joint Development with Sun Microsystems in 1995
- JScript created by Microsoft
- IE and Netscape renderings are slightly different
- Standardized by European Computer Manufacturers Association (ECMA) - <http://www.ecma-international.org/publications/standards/Ecma-262.htm>

Theoretical Background of the Experiment

Uses of JavaScript

- Provide alternative to server-side programming , Servers - often overloaded , Client processing - quicker reaction time
- JavaScript can work with forms
- Event-Driven Computation - a JavaScript program could validate data in a form before it is submitted to a server
- JavaScript can interact with the internal model of the web page (Document Object Model)
- JavaScript is used to provide more complex user interface than plain forms with HTML/CSS can provide

JavaScript – General Format

Directly embedded

```
<!doctype ...>
<html>
  <head>
    <title> Name of web page </title>
    <script type="text/javascript">
      ...script goes here
    </script>
  </head>
  <body>
    ...page body here: text, forms, tables
    ...more JavaScript if needed
  </body>
</html>
```

JavaScript – General Format

Indirect reference

```
<!doctype ...>
<html>
  <head>
    <title> Name of web page </title>
    <script type="text/javascript" src="tst_number.js"/ >
  </head>
  <body>
    ...page body here: text, forms, tables
    ...more JavaScript if needed
  </body>
</html>
```


JavaScript Basic Examples

```
<script>  
    document.write("Hello World!")  
</script>
```

Example

```
<script>  
  x="Hello World!"  
  document.write(x)  
</script>
```

```
<script>  
  x="World"  
  document.write("Hello " +x)  
</script>
```

JavaScript Popup Boxes - alert

- Alert Box
 - An alert box is used if you want to make sure information comes through to the user.
 - When an alert box pops up, the user will have to click "OK" to proceed.

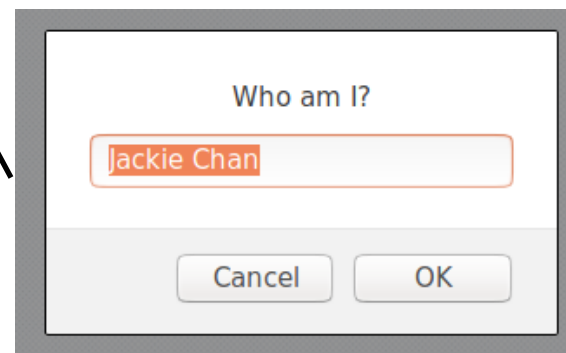
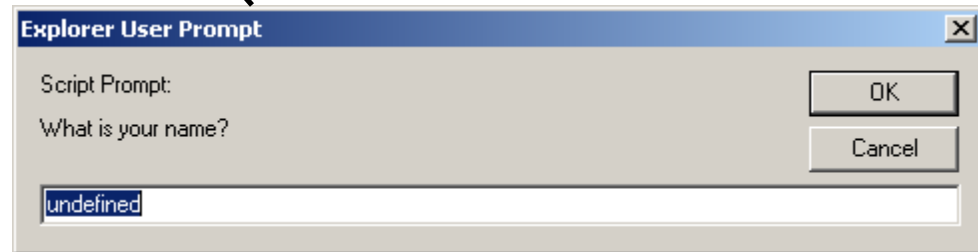
```
<script>  
    alert("Hello World!")  
</script>
```

JavaScript Popup Boxes - confirm

- Confirm Box
 - A confirm box is used if you want the user to verify or accept something.
 - When a confirm box pops up, the user will have to click either "OK" or "Cancel" to proceed.
 - If the user clicks "OK", the box returns true. If the user clicks "Cancel", the box returns false.

alert(), confirm(), and prompt()

```
<script type="text/javascript">  
  alert("This is an Alert method");  
  confirm("Do you want to continue  
this download?");  
  prompt("What is your name?");  
  prompt("Who am I?", "Jackie Chan");  
</script>
```



prompt()

- Return value of prompt() is the value entered in text area
- On clicking **OK** or pressing Enter, the value can be collected in a variable

```
returnValue = prompt("Enter a number");
```

JS Examples -1

Assume : $y = 20x + 12$,
if $x=3$ then, what will be the value of y ?

```
<script>  
  x=3  
  y=20*x+12  
  alert(y)  
</script>
```

Examples -2

```
<script>  
  s1=12  
  s2=28  
  total=s1+s2  
  document.write("sum of two nos.: "+total)  
</script>
```


Statements and Primitive Types

- Statements can be terminated with a semicolon
- But, the interpreter will insert the semicolon if missing at the end of a line
- Can be a problem:

return

x;

- If a statement must be continued to a new line, make sure that the first line does not make a complete statement by itself
- Five primitive types – Number , String , Boolean , Undefined , Null
- Date object

Declaring variable

- Dynamically typed

```
var counter,  
index,  
pi = 3.14159265,  
quarterback = "Elway",  
stop_flag = true;
```

String Catenation +

- +
- Types automatically converted to string
- Implicit , Explicit type conversion

```
document.write( "<br/> counter = " + counter );  
document.write( "<br/> index = " + index );  
document.write( "<br/> pi = " + pi );  
document.write( "<hr/> quarterback = " + quarterback );  
document.write( " stop_flag = " + stop_flag);
```

Conditional Statements

- Similar to C/C++/Java
- **if** statement
- **if...else** statement
- **if...else if...else** statement
- **switch** statement

```
x=3
if( x < 0 )
{
    alert (x + " is negative")
}
else
{
    alert (x + " is positive")
}
```

Looping Statements

- Loop statements in JavaScript are similar to those in C/C++/Java
- while
- for
- do while

```
document.write( "<br/> 1 to 10 ")  
for( i=1; i<10; i++ )  
{  
    document.write( "<br/> " + i )  
}
```

Looping Statements – Print prime numbers

- Prime number - A number that is divisible only by itself and 1
- Using looping statement, find and print prime numbers from 2 to 100

Function and Function call

- Functions are objects in JavaScript

```
function function_name(optional-formal-parameters)
{
    return value;
}
```

```
rvalue = function_name(parameters);
```

Function and Function call

- Functions are objects in JavaScript

```
function fun()  
{  
    document.write( " <hr/> <br/> This surely is fun! <br/>" );  
}
```

```
ref_fun = fun; // Now, ref_fun refers to the fun object
```

```
fun(); // A call to fun
```

```
ref_fun(); // Also a call to fun
```


Fibonacci Numbers

- Fibonacci numbers - integer sequence characterized by the fact that every number after the first two is the sum of the two preceding ones
- 0 , 1 , 1 , 2 , 3 , 5 , 8 . . .
- $F_n = F_{n-1} + F_{n-2}$
- Golden ratio
- Pascal triangle



[Leonardo Bonacci, Leonardo of Pisa, Leonardo Pisano Bigollo, or Leonardo Fibonacci;
Rosen, Kenneth H. Discrete mathematics and its applications / Kenneth H. Rosen.]

1a. Pseudo Code / Outline of the Algorithm

Fibonacci

```
<script type="text/javascript">  
</script>
```

1a. Pseudo Code / Outline of the Algorithm

Fibonacci

```
<script type="text/javascript">  
    var n,a=0,b=1,i,c  
    n=prompt("Enter a number","")  
</script>
```

1a. Pseudo Code / Outline of the Algorithm

Fibonacci

```
<script type="text/javascript">  
  
    var n,a=0,b=1,i,c  
    n=prompt("Enter a number","")  
  
    if(n<0)  
        alert("invalid number")  
  
</script>
```

1a. Pseudo Code / Outline of the Algorithm

```
<script type="text/javascript">

    var n,a=0,b=1,i,c
    n=prompt("Enter a number","")

    if(n<0)
        alert("invalid number")
    else
    {
        if(n==1)
            document.write(a)
        else
            document.write(a+"<br/>" +b)
    }

</script>
```

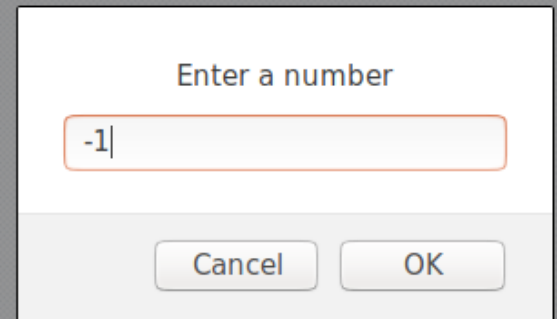
1a. Pseudo Code / Outline of the Algorithm

```
<script type="text/javascript">
    var n,a=0,b=1,i,c
    n=prompt("Enter a number","")
    if(n<0)
        alert("invalid number")
    else
    {
        if(n==1)
            document.write(a)
        else
            document.write(a+"<br/>" +b)

        for (i=2;i<=n;i++) {
            c=a+b
            a=b
            b=c
            document.write("<br/>" +c)
        }
    }
</script>
```

Sample Run

Calculating the fibonacci numbers

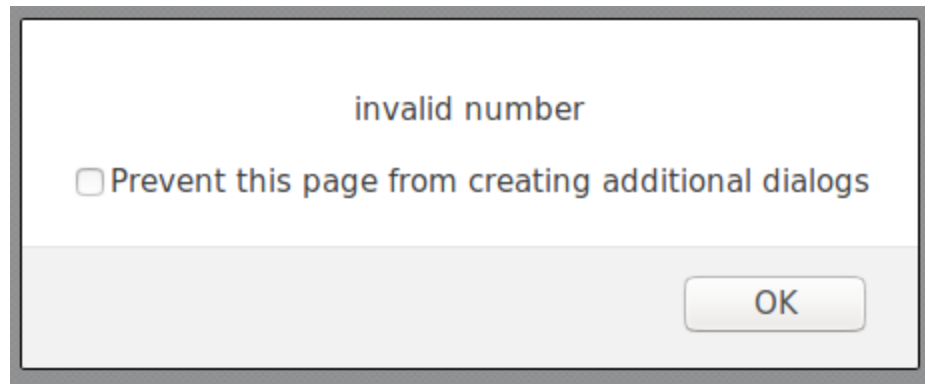


Enter a number

-1

Cancel OK

Sample Run



Sample Run

Calculating the fibonacci numbers

Enter a number

Sample Run

Calculating the fibonacci numbers

0
1
1
2
3
5
8
13
21
34
55

1b. Pseudo Code / Outline of the Algorithm Squares

```
<script type="text/javascript">  
    var n,i  
    n=prompt("Enter a number")  
  
</script>
```

1b. Pseudo Code / Outline of the Algorithm Squares

```
<script type="text/javascript">  
  var n,i  
  n=prompt("Enter a number")  
  
  if(n>0)  
  {  
  
  }  
  else  
    alert("Enter a number greater than one")  
  
</script>
```

1b. Pseudo Code / Outline of the Algorithm Squares

```
<script type="text/javascript">
  var n,i
  n=prompt("Enter a number")

  if(n>0)
  {
    c="Number|Square"
    for(i=1;i<=n;i++)
      c=(c+"\n"+i+"-->" +i*i)
    alert(c)
  }
  else
    alert("Enter a number greater than one")

</script>
```

Sample Run

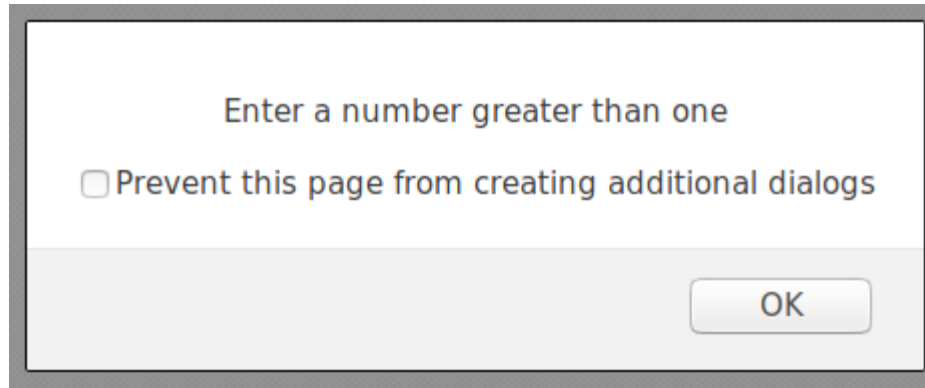
Printing numbers & calculating their squares

Enter a number

-2|

Cancel OK

Sample Run



Enter a number greater than one

☐ Prevent this page from creating additional dialogs

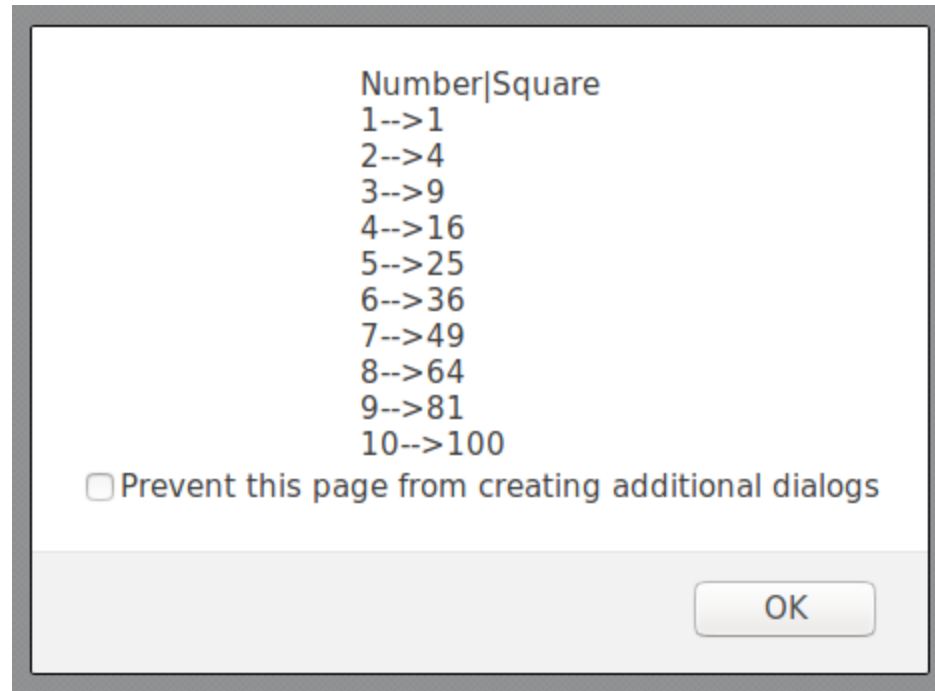
OK

Sample Run

Printing numbers & calculating their squares

Enter a number

Sample Run



Learning Outcomes of the Experiment

At the end of the session, students should be able to :

- 1) Explain the features of Javascript. [L2]
- 2) Experiment with the usage of basic programming concepts like variables, data types and conditional and looping statements [L3]

Firebug



[<http://getfirebug.com/img/firebug-large.png>]