1. **What are literals?**

Answer: The raw data that make up the root of data types are called “literals”. Numbers, strings and Boolean values make up the core set of literals in JavaScript.

1. **What are Hexadecimal Literals?**

Answer: Hexadecimal literals are interpreted as binary string, where each pair of digits represents a character. When used in a numeric context, they are interpreted as integers. Base 16 or Hexadecimal Literals have a special preface that the combination of numbers and letters. All Hexadecimal Literals are prefaced by 0x (zero-x), followed by 0-9, A-F characters indicating a Hexadecimal value. For example color red in Hexadecimal is FF0000; in JavaScript, it is written as 0xFF0000.

1. **What are strings?**

Answers: String literals are text, like any programming languages. Any set of characters placed in a quotation marks (single or double) make up a string literal. Numbers in strings are treated as text, not as value that can be calculated. For example:

“Hello!” ; “25” ; “65 Zigatola” etc.

1. **What do you mean by escape sequence? Write down some escape sequence.**

Answers: Escape sequences includes by prefacing a code with a backslash(\) for additional control over string literals. For example the literal \’ prints an apostrophe without affecting the literal itself. Other escape codes include the following-

\n new line

\’ single quote or apostrophe

\” double quote

\\Backslash

Escape sequences works well with the alert( ) function, it does not work the same with write( ) function. The character substitutions for apostrophes and quotes return the same result but the \n sequence does not rather we’ve to use the <br> to achieve a new line.

1. **What are Boolean values in JavaScript?**

Answer: The Boolean values in JavaScript are two literals, *true* and *false* (1 or 0, yes or no). The Boolean literals are derived from logical comparisons, testing a truth value and then using that value (true or false) for another operation. For example:

<script>

var qpt=((45%2)==0)? "hello" : "bye";

document.write(qpt);

</script>

The output will be: “bye”. Because the statement is false.

1. **Write about concatenations in JavaScript?**

Answer: String concatenation refers to binding one or more strings into a single string. For example:

<script>

var a = “Hello”;

var b = “There!”;

var c = a + “ ” + b;

document.write(c);

</script>

The output will be: Hello There!

1. **What do you mean by object? Write some built in object.**

Answer: Objects are collection of properties arranged in a hierarchy. JavaScript supports working with objects. The highest level of objects in the context of JavaScript and an HTML page is the window. Everything in an HTML page is the property of the window object. There are several useful objects already available in JavaScript, and user can create his own. Objects are used for working with specific data and have different functions which help the user in programming tasks.

Objects are unique in some way, even if two or more objects look identical to the user in the browser. Three very important facets of objects define what it is, what it looks like, how it behaves and how scripts control it. Those three facets are properties, methods and event handlers.

Built in Objects: The built-in objects in JavaScript are Date, Math, String, Array, and Object. Each is used in a unique and not-quite-consistent way.

1. **What do you mean by function and return statement?**

Answer: Function is a definition of a set of deferred actions. Functions are invoked by event handlers or by statements elsewhere in the script. Functions carry out actions and return values. A function is a named set of JavaScript statements interpreted all at once by calling the function name. JavaScript has several built-in functions, but programmer can extend the list by writing his own.

Return Statement: The return statement specifies the value to be returned by a function and performs the act of returning that value to where the function was called from. The following example returns the average of three numbers entered as arguments:   
 Code:   
function average(a, b, c)   
{   
   return (a + b + c)/3;   
}

JavaScript supports "return" statements to allow functions to return values back to calling expressions.

1. **What is undefined and null value?**

Answer: An undefined value is returned when we attempt to use a variable that has not been defined or one that is declared but that we forgot to provide with a value. A nonexistent property of an object also returns undefined if it is addressed.

On the other hand, null amounts to a “nothing literal”. We can declare and define a variable as null if we want absolutely nothing in it but we don’t want it to be undefined. Null is not the same as zero in JavaScript.

1. **What are regular expression literals?**

Answer: Regular expressions are patterns used to match character combinations in strings. In JavaScript, regular expressions are also objects. Regular expressions are used to perform pattern-matching and "search-and-replace" functions on text. For example:

<script>

function x()

{

var s= "Good 100%";

var pattern = /\D/g;

var output= s.match(pattern);

document.write(output);

}

x();

</script>

The output will be: G,o,o,d,%

1. **What are variables?**

Answer: JavaScript variables are containers for storing data values. You can place data into these containers and then refer to the data simply by naming the container. For example;

var x = 5;  
var y = 6;  
var z = x + y;

In here x stores the value 5, y stores the value 6 and z stores the value 11.

1. **What do you mean by Global Variable and Local Variable?**

Answer: Local: These variables only exist inside the specific function that creates them. They are unknown to other functions and to the main program. As such, they are normally implemented using a stack. Local variables cease to exist once the function that created them is completed. They are recreated each time a function is executed or called.

Global: These variables can be accessed (ie known) by any function comprising the program. They are implemented by associating memory locations with variable names. They do not get recreated if the function is recalled.

1. **What is Primitive and Compound Data?**

Answer: Data types are divided into two basic categories, Primitive and Compound. Boolean values, numbers, strings and the null and undefined values all constitute Primitive data types.

Compound data types are made up of more than one component. Two Primitive data types can make up other compound data.

In addition to a multiple of Primitives, Compound data are made up of arrays and objects.

1. **What is Array?**

Answer: An array is a collection of homogenous data elements which are stored in consecutive memory locations. Always address location of array starts from 0. In JavaScript arrays are treated as objects. Each property in an array is called an Element and each element can be assigned a value. An array can be declared in JavaScript in the following ways…

Var myArray=new Array( );

Var myArray=new Array(“Value 1”, “Value 2”, “Value 3”, “Value 4”, “Value 5”);

Var myArray=new Array(dimension argument);

1. **Define Array Length?**

Answer: The single array property Length returns the number of elements in an array. The property is easily passed to a variable. For example:

<script>

var fruits = ["Banana", "Orange", "Apple", "Mango"];

var a = fruits.length;

document.write(a);

</script>

The output will be: 4

1. **What is join(), toString() and concat()?**

Answer: The Array.join() method takes all the values in all of the elements in the array and creates one big string. For example:

var trees=new Array("Elm","Pine","Oak");

var bigBush=trees.join();

document.write(bigBush);

The output will be: Elm,Pine,Oak

The toString() method generates the same results as join(), but we can’t specify the connecting characters between elements as we can with join().

The concat() method not only join all existing elements, but it also adds the elements in a concat() argument. For example:

var trees=new Array("Elm","Pine","Oak");

var biggerBush=trees.concat("Maple","Sycamore");

var bigBush=trees.join(" ");

document.write(biggerBush);

alert(bigBush);

The output will be: An alert box showing “Elm Pine Oak” and then “Elm,Pine,Oak,Maple,Sycamore”