**Datatypes in JS**

1.How many data types are there in JS?

a.2

b.3

c.5

d.7

2.The type of a variable that is volatile is

a. Volatile variable

b. Mutable variable

c. Immutable variable

d. Dynamic variable

3.Mention datatypes in JS?

a. Primitive data type

b.Non-primitive (reference) data type

4. How many Primitive data types are there?

a.3

b.4

c.5

d.6

5.How many Non-primitive (reference) data types are there?

a.2

b.3

c.4

d.5

6.Mention Primitive data types?

String: - represents sequence of characters e.g. "hello"

Number:- represents numeric values e.g. 100

Boolean:- represents boolean value either false or true

Undefined:- represents undefined value

Null:-represents null i.e. no value at all

7.Mention Non-primitive (reference) data types?

Object:-represents instance through which we can access members

Array:-represents group of similar values

8. JavaScript Code can be called by using

a. RMI

b. Triggering Event

c. Preprocessor

d. Function/Method

9. The escape sequence ‘\f’ stands for

a. Floating numbers

b. Representation of functions that returns a value

c. \f is not present in JavaScript

d. Form feed

10. The snippet that has to be used to check if “a” is not equal to “null” is

a. if(a!=null)

b. if (!a)

c. if(a!null)

d. if(a!==null)

**Operators in JS**

1.Assignment Operators is following type of operator \_\_\_\_\_\_\_\_\_\_\_\_\_\_.

A.Ternary

B.None of these

C.Unarry

D.Binary

A.Is identical (is equal to and is of the same type)

B.None of these

C.Is identical (is equal to and is of the different type)

D.Is equal to

3.Which of the following operator is used to concatenate two strings.

A.Comma

B.Arrow

C.Dot

D.Plus

4."++" can operate only on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

A.Constants

B.None of these

C.Variables

D.Text

5."++" is \_\_\_\_\_\_\_\_\_\_ type of operator.

A.None of these

B.Ternary

C.Unary

D.Binary

6.Integer Variable + Character Variable + Undefined Variable = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Value

A.Character

B..Integer

C.Undefined Value

D.Text / String Value

7.Which of the following is not a comparison operator ?

A.==

B.>=

C.<=

D.++

E.===

8.Adding String and Integer always results in \_\_\_\_\_\_\_\_\_.

A.Integer

B.String

C.Character

D.None of these

9.JavaScript contains a \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ that assigns a value to a variable based on some condition.

A.Bitwise Operator

B.Assignment operator

C.Logical Operator

D.Conditional operator

10.Conditional Operator shown in the following example is \_\_\_\_\_\_\_\_\_\_\_\_\_.

marks = (mark<35)?"Fail":"Pass";

A.Ternary Operator

B.Unary Operator

C.Binary Operator

D.none of these

**Variable in JS**

1.Variable can hold \_\_\_\_\_\_\_\_ value at a time

A.Multiple

B.None of these

C.Double

D.Single

2.Integer Variable is declared using following syntax in JavaScript.

**A.var** num;

B.**Integer** num;  
C.**integer** num;  
D.**int** num;

3.We can declare **\_\_\_\_\_\_\_\_\_\_\_** at a time. Select most appropriate option.

A.One or more Variables

B.More than One Variables

C.One Variable Only

D.None of these

4.We can declare all type of variables in JavaScript with the keyword \_\_\_\_\_\_\_\_\_\_\_\_\_.

A.None of these

B.var

C.jvar

D.obj

5.**var** num;  
Above Statement is called as \_\_\_\_\_\_\_\_\_\_ Statement.

A.Initialization

B.Instantization

C.Declaration

D.Globalization

6.Spaces,Punctuation marks are called as \_\_\_\_\_\_\_\_\_\_ Symbols in JavaScript.

A.Mandatory

B.Special

C.None of these

D.Punctual

7.Which JavaScript variable cannot be used as First character but can be use after first character ?

A.Asterisk

B.Underscore

C.Digit

D.Dollar Sign

8.Java Script Variable should be Case \_\_\_\_\_\_\_\_\_\_\_.

A.Sensitive

B.Non Sensitive

9.Underscore can be used as first letter while declaring variable in JavaScript .

A.False

B.True

9.Multiple Declarations of variables are separated by \_\_\_\_\_\_\_\_\_\_\_ symbol

A.Semicolon

B.Asterisk

C.Colon

D.Comma

**Objects in JS**

1. The unordered collection of properties, each of which has a name and a value is called

a. String

b. Object

c. Serialized Object

d. All of the above

2. The object has three object attributes namely

a. Class, parameters, object’s extensible flag

b. Prototype, class, objects’ parameters

c. Prototype, class, object’s extensible flag

d. Native object, Classes and Interfaces and Object’s extensible flag

3.Consider the following code snippet :

var book = {  
"main title": "JavaScript",   
'sub-title': "The Definitive Guide",   
"for": "all audiences",   
author: {   
firstname: "David",   
surname: "Flanagan"   
}  
};

In the above snippet, firstname and surname are

a. properties

b. property values

c. property names

d. objects

4. A linkage of series of prototype objects is called as :

a. prototype stack

b. prototype chain

c. prototype class

d. prototypes

5.. Consider the below given syntax

book[datatype]=assignment\_value;

In the above syntax, the datatype within the square brackets must be

a. An integer

b. A String

c. An object

d. None of the mentioned

6. To determine whether one object is the prototype of (or is part of the prototype chain of) another object, one should use the

a. isPrototypeOf() method

b. equals() method

c. === operator

d. None of the mentioned

7. Consider the following code snippet

function f() {};

The above prototype represents a

a. Function f

b. A custom constructor

c. Prototype of a function

d. Not valid

8.8. The purpose of *extensible* attribute is to

a. make all of the own properties of that object nonconfigurable

b. to configure and bring a writable property

c. “lock down” objects into a known state and prevent outside tampering

d. All of the mentioned

9. Identify the process done in the below code snippet

o = {x:1, y:{z:[false,null,""]}};   
s = JSON.stringify(o);   
p = JSON.parse(s);

a. Object Encapsulation

b. Object Serialization

c. Object Abstraction

d. Object Encoding

10. The basic purpose of the toLocaleString() is to

a. return a localised object representation

b. return a parsed string

c. return a local time in the string format

d. return a localized string representation of the object

**Arrays in JS**

1.Consider the code snippet given below

var count = [1,,3];

What is the observation made?

a. The omitted value takes “undefined”

b. This results in an error

c. This results in an exception

d. None of the mentioned

2. Consider the following code snippet

var a1 = [,,,];   
var a2 = new Array(3);   
0 in a1   
0 in a2

The result would be

a. true false

b. false true

c. true true

d. false true

3. The pop() method of the array does which of the following task ?

a. decrements the total length by 1

b. increments the total length by 1

c. prints the first element but no effect on the length

d. None of the above

|  |  |  |
| --- | --- | --- |
|  | 4. Consider the following code snippet :  if (!a[i]) continue;  What is the observation made ?  a. Skips the undefined elements  b. Skips the non existent elements  c. Skips the null elements  d. All of the mentioned  5. What will happen if reverse() and join() methods are used simultaneously ?  a. Reverses and stores in the same array  b. Reverses and concatenates the elements of the array  c. Reverses  d. All of the mentioned  6. Consider the following code snippet :  var a = [1,2,3,4,5]; a.slice(0,3);  What is the possible output for the above code snippet ?  a. Returns [1,2,3] b. Returns [4,5] c. Returns [1,2,3,4] d. Returns [1,2,3,4,5]  7. Consider the following code snippet :  var a = [];  a.unshift(1);  a.unshift(22); a.shift();  a.unshift(3,[4,5]);  a.shift();  a.shift(); a.shift(); |  |
|  | The final output for the shift() is  a. 1  b. [4,5] c. [3,4,5] d. Exception is thrown  8. The primary purpose of the array map() function is that it  a. maps the elements of another array into itself  b. passes each element of the array and returns the necessary mapped elements  c. passes each element of the array on which it is invoked to the function you specify, and returns an array containing the values returned by that function.  d. None of the mentioned  9. The reduce and reduceRight methods follow a common operation called  a. filter and fold  b. inject and fold  c. finger and fold  d. fold  10. The method or operator used to identify the array is  a) isarrayType()  b) ==  c) ===  d) typeof |  |
|  | **Strings in JS**  1.What is the correct JavaScript syntax to write "Hello World"?  A. System.out.println("Hello World")  B. println ("Hello World")  C. document.write("Hello World")  D. response.write("Hello World")    2. Which of the following way can be used to indicate the LANGUAGE attribute?  A. <LANGUAGE="JavaScriptVersion">  B. <SCRIPT LANGUAGE="JavaScriptVersion">  C. <SCRIPT LANGUAGE="JavaScriptVersion"> JavaScript statements…</SCRIPT>  D. <SCRIPT LANGUAGE="JavaScriptVersion"!> JavaScript statements…</SCRIPT>    3. Inside which HTML element do we put the JavaScript?  A. <js>  B. <scripting>  C. <script>  D. <javascript>  4. What is the correct syntax for referring to an external script called " abc.js"?  A. <script href=" abc.js">  B. <script name=" abc.js">  C. <script src=" abc.js">  D. None of the above |  |
|  | 5.JavaScript entities start with \_\_\_\_\_\_\_ and end with \_\_\_\_\_\_\_\_\_.  A. Semicolon, colon  B. Semicolon, Ampersand  C. Ampersand, colon  D. Ampersand, semicolon    6.’\f’ this symbol is used for  A.carriage  B return  C.backspace  D.form feed  7.The statement a===b refers to  A. Both a and b are equal in value, type and reference address  B. Both a and b are equal in value  C. Both a and b are equal in value and type  D. There is no such state    8. Assume that we have to convert “false” that is a non-string to string. The command that we use is (without invoking the “new” operator)  A. false.toString()  B. String(false)  C. String new variable=”false”  D. Both a and b  9. Which of the following is not considered as an error in JavaScript?  A. Syntax error  B. Missing of semicolons  C. Division by zero  D. All of the above  10.When there is an indefinite or an infinity value during an arithmetic value computation, javascript  A. Prints an exception error  B. Prints an overflow error  C. Displays “Infinity”  D. Prints the value as such  **Numbers and Boolean in JS** 1: Javascript is \_\_ language.  * Loosely Typed * Strongly Typed * Dynamic * Both Loosely Typed and Dynamic   2: Which of the following is a JavaScript datatype?   * Null * Object * Undefined * All of them   3: Which of the following is not a JavaScript datatype?   * Boolean * Number * String * Function   4: typeof NaN (Not a Number)   * Number * String * NaN * Object  5: typeof null  * Number * String * NaN * Object       6: How does JavaScript store dates in objects of Date type?   * The number of days since 1st, 1900 * The number of seconds since January 1st, 1970 * The number of milliseconds since January 1st, 1970 * The number of pico seconds since January 1st, 1970  7: The \_\_ value is returned when you use an object property that does not exist, or a variable that has been declared, but has never had a value assigned to it.  * null * undefined * NaN * None of the above     8: In Javascript, number data types are stored in \_\_ floating point format   * double-precision 64-bit * single-precision 32-bit   9: typeof Infinity   * Number * String * NaN * Object  10: What is the output? x = 100 / "Mango";  * null * undefined * NaN * Infinity     **Functions in JS**  1: What is the use of a return statement in a function?  * Returns the value and continues executing rest of the statements * Returns the value and stops the program * Returns the value and stops executing the function * Stops executing the function and returns the value   2: Is it valid to nest functions in JavaScript?   * Yes * No   3: What is an array-like object containing the arguments passed to the currently executing function.   * arguments * function * args * None of the above   4: Which of the following are different ways to create a function?   * function [name]([param1] [, param2] [..., param3]){ statements } * var [name] = function ([param1] [, param2] [..., param3]){ statements * new Function (arg1, arg2, ... argN, functionBody) * All of the above  5: How to determine whether a function exists by using the typeof operator?var square = function(number1, number2) { return number1 + number2 };  * typeof square == 'function' * typeof square == 'object'   6: If you pass less number of parameters than the expected number of parameters to a function, the missing parameters get assigned the value \_\_.   * undefined * null * Syntax Error * None of the above  7: How to create a function with the JavaScript Function contructor ?  * var func = Function("x","y","return x+y"); * var func = Function(x,y){ return x+y;} * var func = new Function("x", "y", "return x + y"); * None of the above   8: Which of the following javascript functions are used to convert non numeric values into numbers?  * Number() * parseInt() * parseFloat() * All of the above   9: When does the function name become optional in JavaScript?  * Function is defined within a function * Function is defined using Function constructor * Function is defined as an expression * Function is predefined  10: What will be output of the following code snippet?var result = (function(x) {return x\*x;}(10));  * 10 * 100 * 20 * None of the above   **Local and Global scope of variable** |  |
|  | 1. The function definitions in JavaScript begins with  a. Identifier and Parantheses  b. Return type and Identifier  c. Return type, Function keyword, Identifier and Parantheses  d. Identifier and Return type  2. Consider the following code snippet  function printprops(o)  {  for(var p in o)  console.log(p + ": " + o[p] + "\n"); }  What will the above code snippet result ?  a. Prints the contents of each property of o  b. Returns undefined  c. Both a and b  d. None of the mentioned  3. When does the function name become optional in JavaScript?  a. When the function is defined as a looping statement  b. When the function is defined as expressions  c. When the function is predefined  d. All of the mentioned  4. What is the purpose of a **return** statement in a function?  a. Returns the value and continues executing rest of the statements, if any  b. Returns the value and stops the program  c. Returns the value and stops executing the function  d. Stops executing the function and returns the value  5. What will happen if a return statement does not have an associated expression?  a. It returns the value 0  b. It will throw an exception  c. It returns the **undefined** value  d. None of the mentioned  6. A function with no return value is called  a. Procedures  b. Method  c. Static function  d. Dynamic function  7. Consider the following code snippet  function hypotenuse(a, b)  {  function square(x)   {   return x\*x;   }  return Math.sqrt(square(a) + square(b)); }  What does the above code result?  a. Sum of square of a and b  b. Square of sum of a and b  c. Sum of a and b square  d. None of the mentioned  8. Which of the following is the correct code for invoking a function without **this** keyword at all, and also too determine whether the strict mode is in effect?  a. var strict = (function { return this; }); b. mode strict = (function() { return !this; }()); c. var strict = (function() { return !this; }()); d. mode strict = (function { });  9. Which is an equivalent code to invoke a function **m** of class **o** that expects two arguments x and y?  a. o(x,y); b. o.m(x) && o.m(y); c. m(x,y); d. o.m(x,y);  10. Consider the following code snippet  o.m(x,y);  Which is an equivalent code for the above code snippet?  a. o.m(x) && o.m(y); b. o["m"](x,y); c. o(m)["x","y"]; d. o.m(x && y);  **Code Example for Object**  1.var rabbit = {}; rabbit.speak = function(line) {  print("The rabbit says '", line, "'"); }; rabbit.speak("Well, now you're asking me.");  2.function speak(line) {  print("The ", this.adjective, " rabbit says '", line, "'"); } var whiteRabbit = {adjective: "white", speak: speak}; var fatRabbit = {adjective: "fat", speak: speak}; whiteRabbit.speak("Oh my ears and whiskers, how late it's getting!"); fatRabbit.speak("I could sure use a carrot right now.");  3.function Rabbit(adjective) {  this.adjective = adjective;  this.speak = function(line) {  print("The ", this.adjective, " rabbit says '", line, "'");  }; } var killerRabbit = new Rabbit("killer"); killerRabbit.speak("GRAAAAAAAAAH!");  4.Object.prototype.properties = function() {  var result = [];  for (var property in this)  result.push(property);  return result; };  var test = {x: 10, y: 3}; show(test.properties());  5.function forEachIn(object, action) {  for (var property in object) {  if (object.hasOwnProperty(property))  action(property, object[property]);  } } var chimera = {head: "lion", body: "goat", tail: "snake"}; forEachIn(chimera, function(name, value) {  print("The ", name, " of a ", value, "."); });  **Code Example for Arrays**  1.function show\_array(array) {  for(var i=0; i<array.length; i++) {  document.write(array[i]);  document.write('<br/>');  } } var fruits = ['apple', 'orange', 'banana']; show\_array(fruits);  2.var cat = {colour: "grey", name: "Spot", size: 46}; cat.size = 47; show(cat.size); delete cat.size; show(cat.size); show(cat);  3.var thing = {"gabba gabba": "hey", "5": 10}; show(thing["5"]); thing["5"] = 20; show(thing[2 + 3]); delete thing["gabba gabba"];  4.var object1 = {value: 10}; var object2 = object1; var object3 = {value: 10};  show(object1 == object2); show(object1 == object3);  object1.value = 15; show(object2.value); show(object3.value);  5.var mailArchive = {0: "Dear nephew, ... (mail number 1)",  1: "(mail number 2)",  2: "(mail number 3)"};  for (var current = 0; current in mailArchive; current++)  print("Processing e-mail #", current, ": ", mailArchive[current]);  **Code Example for Function**  1.function forEach(array, action) {  for (var i = 0; i < array.length; i++)  action(array[i]); }  forEach(["Wampeter", "Foma", "Granfalloon"], print);  2.show(Math.min.apply(null, [5, 6]));  function negate(func) {  return function() {  return !func.apply(null, arguments);  }; }  3.unction map(func, array) {  var result = [];  forEach(array, function (element) {  result.push(func(element));  });  return result; }  show(map(Math.round, [0.01, 2, 9.89, Math.PI]));  4.function compose(func1, func2) {  return function() {  return func1(func2.apply(null, arguments));  }; }  var isUndefined = partial(op["==="], undefined); var isDefined = compose(op["!"], isUndefined); show(isDefined(Math.PI)); show(isDefined(Math.PIE));  5.var url = "http://www.gokgs.com/"; var text = "Play Go!"; var linkText = "<a href=\"" + url + "\">" + text + "</a>"; print(linkText);  **Code Example for Local and Global Variable**  1.<script>  // Global variable.  var a = "Dot Net Tricks !";  function Show() {  // A local variable is declared in this function.  var a = "Hello World !";    alert("Value of 'a' inside the function " + a); //Hello World !  }    alert("Value of 'a' outside the function : " + a); //Dot Net Tricks !  2.<script >  // Global variable.  var a = "Dot Net Tricks !";  function Show() {  // A Local variable is declared in this function.  var a = "Hello World !";  alert("Value of 'a' inside the function :" + a); //Hello World !    //b will have global scope  b = "Hello JavaScript !";  Display();  }    alert("Value of 'a' outside the function : " + a); //Dot Net Tricks !    function Display() {  //Since b variable has global scope  alert("Value of 'b' outside the function : " + b); //Hello JavaScript !  }  3.var aNumber = 100; tweak(); function tweak(){  document.write(aNumber);   if (false)  {  var aNumber = 123;   } }  4.let x = 10; var y = 10; {  let x = 5;  var y = 5;  {  let x = 2;  var y = 2;  document.write("x: " + x + "<br/>");  document.write("y: " + y + "<br/>");  }  document.write("x: " + x + "<br/>");  document.write("y: " + y + "<br/>");  }  document.write("x: " + x + "<br/>"); document.write("y: " + y + "<br/>"); Output:-  x: 2  y: 2  x: 5  y: 2  x: 10  y: 2  5.var aCentaur = "a horse with rider,"; function antiquities(){   var aCentaur = "A centaur is probably a mounted Scythian warrior"; }  antiquities(); aCentaur += " as seen from a distance by a naive innocent."; document.write(aCentaur); |  |

|  |
| --- |
|  |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |