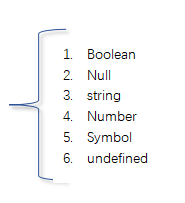
**a.**



JavaScript has six primitive data types.

1. The types of primitive types supported in the language are Boolean which is a value for holding true or false.

Eg: let b=true; // b now has a type boolean

2. Null ( value of null )

Eg: let n=null; // n now has a type null

3. String which is essentially set of a character array.

Eg：let s= “hello world”//s now has a type string

4. Number which is a numeric format which is internally a floating-point representation but it can be used to hold integers or floating-point numbers.

Eg: let s=3.14//s now has a type number

5. Symbol which is an atomic immutable type. the variables value can be changed after it is set in the first place.

Eg: let id=Symbol("id"); //id now has a type symbol

6. Undefined( value of undefined variables and constants)

  Eg: let  s; // s now has a type undefined

**b.**

1. global variables(If undefined, output  " undefined "): var

2.**block variables** (If undefined, the compiler will yell at you.)**:**

1) Let : A variable is defined by the LET keyword its value can be changed later.

2) Const: A variable is defined by the CONST keyword its value can not be changed later.

**C.**

1) What is a closure ?

A common use case for this is called a closure where the return function closes over but still has access to a constant or a variable which would otherwise go out of scope.

2) Consider the example here the function 'make adder' it takes an argument x and returns a function which returns a function which takes a value Y and returns the sum of those. In fact 'make adder' is a function which closes over the variable x and its values.

**d.**

1) Object methods

As objects’ most basic an object is a data structure for holding key value properties, and these properties can be primitive, they can be themselves objects or they can be functions. They have a property, a value, a comma, another property, a value and so on.

2) How do object methods refer to object properties?

We can address elements within the properties and properties within the objects using the dot syntax as you can see here the 'object.name' accesses the name in the object or we can also use the sort of arrays subscripting syntax. Using a string as the property name to get access to that variables that an object's property.

Eg:

Let obj={

Name = “hello”,

Age=23 };

Console.log ( obj.Name , obj [“ Age”] );

**e.**

1) JavaScript arrays essentially allow mixed types to be held within the same array.

Eg : let arr = [ “hello” , 1, true ] ;

2) You don't need to pre-declare the array size as the arrays length will adjust automatically as elements are added.

Eg : let arr =[ “internet” , 5, “end” ] ;

3) Arrays can be defined using an array literal syntax and accessed in the familiar way with numeric indexing.

Eg: let arr =[ “internet” , 5, “end” ] ;

arr [0]  //  “internet”

arr [1]   //  5

4) we can also assign the elements using some subscripting syntax shown.

Eg: let arr =[ “internet” , 5, “end” ] ;

arr [0]  //  “internet”

arr[1] = 6 ;

arr[1]   //   6

5) And as expected array indexing is zero based meaning that the first element of the array is at index 0.

Eg: let arr =[ “teacher” , 5 , “end” ] ;

arr [0]  //  “teacher”