Node JS Module – 1

1) **What is the difference between Java & JavaScript?**

ANS : Java Is a object oriented language is it use buliding application and JavaScript is a dynamically Create a web page it's run in every web browser.

2) **What Is JavaScript ?**

Ans : JavaScript is high-level, interpreted language primary used to make web pages interactive and dynamic. It was initially created by Brendan Eich in 1995.

3) **Whar are the data types supported by JavaScript ?**

Ans : in the JavaScripte supported primitive and non-primitive data types.

Here list primitive data types.

1) Number
2) String
3) Boolean
4) Undefiled
5) Null
6) Symbol.

4) **What are the scopes of variable in JavaScript ?**

ANS : There are two scope of variable Global Scope And Local Scope

Global Scope : in that the variable can be accessed anywhere in thr program is known as with global scope.

Local Scope : in this variable that you declare inside of function is a local scope. You can access a local variable can within a function.

5) **What is Callback ?**

Ans : A callback in JavaScript is a function that you can pass an argument to another function. This function gets executed later, after some time other code or function finishe running, it like giving instructions to call you back once a certain task is done.

6) **What is Closure ? Give An Example.**

Ans : A closure in javascript is like a backpack That function carries around. This backpack contains all variable that were present in the function's scope when it was created. After the function has finish run, it still access ans use those variable.

Closure Example :

```
function outerFunction(x)
{
        //Inner function is defined inside the outer function.
        Function innerFunction(y)
{
 return x + y;
// innerFunction has access to the 'x' variable from outerFunction
}
  Return innerFunction;
}

//create a closure by calling outerFunction with a value.
Const closure = outerFunction(5);

//use the closure to add a value to the original 'x'
Const result = closure(25);
//Result is 30 (5 + 25)
```

7) What is the difference between the operators '==' & '===' ?

ANS : in javaScript , == and === between comparison operators used to check equality between two values or variable.

Equality Operator(==) :

The == operator check for equality of values after performing type coercion if necessary.for example 1=="1" will return true because javaScript coerces "1" to anumber before comparison.

Identity Operator (===) :

The === operators check for equality of values without performing type coercion. It strictly checks both the value and the type of operands. For example, 1==='1' will reyurn false because they are different types.

8) What is the difference between Null & undefined ?

ANS : In javascript null and undefined are both values used to represent absence of value but they have slightly different meanings.

Null :

It is a deliberate assignment that represents the It is often used to explicitly indicate that a variable or object property should have no value or no reference to any object absence of any object value. It is also a primitive value. It is defined undefined value and null value in briefly explain.

Undefined :

When a variable is declared but not a word Initialized, or when a function does not return a value, the variable or the function's result is undefined. Accessing an object property or array element that does not exist also results in undefined. It is a primitive value.

9) What would be the result of 2+5+”3”?

ANS : in javaScript 2+5+”3” is 10, 253 or 73 here 73 is right answer because compiler frist do 2+5 as an integer or a addition symbol then “3” is treated as string and + is treated a concat symbol. That the right answer is 73.

10) What is the difference between Call & Apply ?

ANS : in javaScript call and apply are both methods used to invoke function. They are used to call function with a specified “this” value and argument.

Call :

The call method invokes a function with a specified “this” value and individual arguments.

SYNTAX : function.call(thisArg, arg1, arg2, ….);

Apply :

The apply method is similar to call(), but it accepts the function arguments as an array.

SYNTAX : function.apply(thisArg