1. The ” var” and “function” are known as _____.

a. Data types

b. Keywords

c. Prototypes

d. Declaration statements *
Answer:(D)
Reason: The "function" and "var" both are the Declaration statements. These both are used for defining, and declaring variable, function in anywhere in the program.

2. Which of these is a correct output for the JavaScript code given below?

```
string X= “Hey”;

string Y=”There”;

alert(X+Y);
```

a. Hey There

b. Hey_There

c. HeyThere *

d. undefined
Answer: C
Reason: The alert method is commonly used for Displaying the value(or message) passed as argument in the "dialogbox" in the web-browser. Here, the alert method concatenates both given strings and prints as a single string in the form of output.

3. Which of these is known as the Equality operator used for checking whether both the values are equal?

a. =

b. == *

c. ===

d. &&
Answer: B
Reason: **== operator** is a type of Relational Operator in JavaScript used to check for relations of equality. It returns a boolean result after the comparison and is extensively used in looping statements and conditional if-else statements.

```
4. function sayHi() {
  console.log(name);
  console.log(age);
  var name = 'Lydia';
  let age = 21;
}

sayHi();
```
Answer : D
A: Lydia and undefined
B: Lydia and ReferenceError
C: ReferenceError and 21
D: undefined and ReferenceError *

```
5. for (var i = 0; i < 3; i++) {
    setTimeout(() => console.log(i), 1);
  }

  for (let i = 0; i < 3; i++) {
    setTimeout(() => console.log(i), 1);
  }
```
A: 0 1 2 and 0 1 2
B: 0 1 2 and 3 3 3
C: 3 3 3 and 0 1 2 *
Answer: C
Reason: Here “var” keeps the latest value of ‘i’ so it prints three times 3 Whereas “let” binds to inner scope each time differently.

```
6.function Person(firstName, lastName) {
  this.firstName = firstName;
  this.lastName = lastName;
}

const lydia = new Person('Lydia', 'Hallie');
const sarah = Person('Sarah', 'Smith');

console.log(lydia);
console.log(sarah);
Answer: A
Reason: Once the value of the const is declared it cannot be changed again

A: Person {firstName: "Lydia", lastName: "Hallie"} and undefined *
B: Person {firstName: "Lydia", lastName: "Hallie"} and Person {firstName:
"Sarah", lastName: "Smith"}
C: Person {firstName: "Lydia", lastName: "Hallie"} and {}
D:Person {firstName: "Lydia", lastName: "Hallie"} and ReferenceError


7.function sum(a, b) {
    return a + b;
  }

  sum(1, '2');
A: NaN*
B: TypeError
C: "12"
D: 3
Answer:A
Reason: Here return value is not stored any where, so the result will be null


8. let number = 0;
console.log(number++);
console.log(++number);
console.log(number);
Answer: C
Reason: (i++) it is post increment, means it increments the value after execution
of statement. (++i) it is pre-increment, means it increments the value before
execution of statement.
A: 1 1 2
B: 1 2 2
C: 0 2 2*
D: 0 1 2
```

```
9. var num = 8;
var num = 10;

console.log(num);
A: 8
B: 10*
C: SyntaxError
D: ReferenceError
Answer: B
Reason: ‘var’ it takes the latest value



10. const obj = { a: 'one', b: 'two', a: 'three' };
console.log(obj);
A: { a: "one", b: "two" }
B: { b: "two", a: "three" }
C: { a: "three", b: "two" }*
D: SyntaxError
Answer: C

11. Output of code
var text = 'outside';
function logIt(){
    console.log(text);
    var text = 'inside';
};
logIt();
Answer: Undefined
Reason:   Here introducing a new text variable that is local  and due to variable
hoisting, the actual var text line is interpreted before the call to console.log,
hence the  result is undefined.

14.Explain Hoisting with example.
Answer:Hoisting is a kind of default behavior in which all the declarations
either variable declaration or function declaration are moved at the top of the
scope just before executing the program's code.
Example: var place = 'outside';
function logIt(){
    console.log(place);
    var place = 'inside';
};
logIt();
```

15. Output of this code

```
var length = 10;
function fn() {
    console.log(this.length);
}

var obj = {
  length: 5,
  method: function(fn) {
    fn();
    arguments[0]();
  }
};

obj.method(fn, 1);
```

Answer: 10,2

16. what would following code return

```
console.log( typeof 1);
```

Answer: number