Answers

**1.Ans:** In js, we can declare variables in 3 ways.like

             1.const

              2.var

              3.let

**const:** The const declaration declares block-scoped local variables. The value of a constant can't be changed through reassignment using the [assignment operator](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Assignment), but if a constant is an [object](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Data_structures#objects), its properties can be added, updated, or removed.

**var:** On the other hand, The var statement declares function-scoped or globally-scoped variables.In a script, a variable declared using var is added as a non-configurable property of the global object. This means its property descriptor cannot be changed and it cannot be deleted using [delete](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/delete). JavaScript has automatic memory management, and it would make no sense to be able to use the delete operator on a global variable.Duplicate variable declarations using var will not trigger an error, even in strict mode, and the variable will not lose its value, unless the declaration has an initializer.

**let:** And the let declaration declares re-assignable, block-scoped local variables.let declarations are scoped to blocks as well as functions.let declarations do not create properties on [globalThis](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/globalThis) when declared at the top level of a script.

**2. Ans:** Truthy values are expressions that evaluate to the boolean value of true, and falsy values are expressions that evaluate to the boolean value of false. I have included a table with expressions that are falsy.

Falsy expression: False,NaN,undefined,Null,0,Empty String.

Truthy expression: True,numbers which are greater than zero,Non-empty sequences or collections (lists, tuples, strings, dictionaries, sets) etc.

**3.Ans:** There are many types of functions in js.like

               1.Arrow function

                2.Invoked function.

                3.Anonymous function

                4.Constructor function.

                5.Factory function. etc

**4.Ans:** For a typical function, the value of this is the object that the function is accessed on. In other words, if the function call is in the form obj.f(), then this refers to obj. The value of this always changes based on how a function is called. In non–strict mode, this is always a reference to an object. In strict mode, it can be any value.

Example:

 function getThis() {

  return this;

}

const obj1 = { name: "obj1" };

const obj2 = { name: "obj2" };

obj1.getThis = getThis;

obj2.getThis = getThis;

console.log(obj1.getThis());

Advantages of arrow function in methods: To easily read and understand as well as write the code more effectively.

Disadvantages:

1. Arrow functions cannot be used as constructors.

  2.Should not be used as methods.

**5.Ans:** We use return whenever we want to use a function as a variable.

**6.Ans:** The latest ECMAScript standard defines eight data types Out of which seven data types are Primitive(predefined) and one complex or Non-Primitive**.**

1.Primitive Data  Types:  Number,String,Undefined,Symbol,Boolean etc

2. Non-Primitive Data Types:  Object.