1.What are the ways to create the objects?

- **using an object literal**
- **with the keyword new**
- **Using an object Constructor**

Using an object literal, we will define and create an object in one statement.

```
var person = {firstName:"Jagan Reddy", lastName:"Baddam", age:25, eyeColor:"Black"};
```

Using new keyword,

```
var person = new Object();
person.firstName = "Jagan Reddy";
person.lastName = "Baddam";
person.age = 25;
person.eyeColor = "black";
```

Using an object Constructor

```
function person(first, last, age, eye) {
   this.firstName = first;
   this.lastName = last;
   this.age = age;
   this.eyeColor = eye;
}
var person1 = new person("Jagan Reddy", "Baddam", 25, "black");
var person2 = new person("Vinay Reddy", "Sevella", 26, "black");
```

2.How many ways we can create the arrays?

- Using an array literal
- with the keyword new

Using an array literal, we will define and create an Array in one statement.

```
var cars = ["Cadillac", "Tesla", "Ferrari"];
```

with the keyword new

```
var cars = new Array("Cadillac", "Tesla", "Ferrari");
```

3.What are arguments in java script functions?

Arguments are the parameters in java script functions, which are real values received by the function when it is invoked. Inside the function they behave as local variables

```
function name (parameter1, parameter2, parameter3) {
  code to be executed
}
```

4.What is prototypal inheritance in java script?

All java script objects inherit their properties and methods from their prototype. For instance, objects created with new Date () inherit the Date. prototype

The Object. prototype is on the top of the prototype chain

Eg:

```
function Person (first, last, age, eyecolor) {
  this.firstName = first;
```

```
    this.lastName = last;
    this.age = age;
    this.eyeColor = eyecolor;
}
```

now adding a new property in person object using a prototype

```
Person.prototype.nationality = "English";
```

5.What are enumerators in java script?

Enumerators are json like object in js except the properties' values are not in quote. Enum properties can be accessed using dot (.) notation like in json.

Eg:

```
var SizeEnum = {
  SMALL: 1,
  MEDIUM: 2,
  LARGE: 3,
};
```

Then use it like so:

```
var mySize = SizeEnum.SMALL;
```

6.Callbacks and Closures?

Callbacks are the function that contains parameters as another function

A closure is a function having access to the parent scope, even after the parent function has closed.

Eg:

```
function bindEvents() {

    var countryDdl = controls().countries;

    countryDdl.addEventListener("change", function () {

        buildStates(this.value).us();

    });

}
```

```
var add = (function () {
   var counter = 0;
   return function () {return counter += 1;}
})();
```

7.Module based programming in java script.

Java script support the oop concept so we can call it module based programming. Each component can be defined separately and can be used in multiple places in the code. So, the reusability of the component implies that it is modular.

8.What is strict mode in javascript?

The purpose of "use strict" is to indicate that the code should be executed in "strict mode".

With strict mode, you can not, for example, use undeclared variables.

Eg:

```
“use strict”

X = 3.14; // this will cause an error b/c x is not defined
```

What is the difference between == and ===

== equality check, === checks equality and value type also

```
var a = 5;

var b = "5";

var c = (a==b); // true

var d = (a===b); // false
```

9.Ternary operator

The ternary (?) operator can be used as a shortcut for an if...else statement

Eg:

```
var now = new Date();
var greeting = "Good" + ((now.getHours() > 17) ? " evening." : " day.");
```

10.Difference between public, private and static variables and their use cases

Public: Any function can access, modify, or delete those members, or add new members.

```
                    Eg:              function Container(param) {
                    this.member = param;
                    }
      var container = new Container(“a”);
  // now member can be accessed like this:
          Container.member // returns a
```

Private: only private function can access them

Eg:

```
function Container(param) {
var a = 5;
// now a is not accessible using //Container.a, however, private function can access it.

Function add(){
// now add can access a
Var sum = a + 5;
}
}
```

Static: static variables are shared by all instances of the object, useful scenario would be counting example

Eg:

```
function Person(){
  this.name = "Peter";
// counter is a static variable
  Person.counter = Person.counter + 1;


}
Person.counter = 0;
var p1 = new Person;
var p2 = new Person;
console.log(Person.counter); // 2
```