|  |  |  |
| --- | --- | --- |
| JAVASCRIPT | | |
| 1 | **Data types and variable** | |
| 1 | Data Types |  |
| 2 | Primitive & Reference Data Types |  |
|  |  |  |
|  |  |  |
| 2 | **Objects** | |
| 1 | Object Literal |  |
| 2 | Object Methods |  |
| 3 | Constructor |  |
| 4 | Object.create( ) |  |
| 5 | Factory |  |
| 6 | Prototype |  |
| 7 | Method Overloading |  |
|  | Functions overloading or method overloading means defining multiple functions with same name but different parameter(no. of params). eg. myfunction( ) and myfunction(argument) are two different functions and both will work differently based on parameter passed.  Javascript doesn’t support function overloading. if multiple functions are defined with same name, the fuction defined at last will override the previous ones |  |
| 8 | Method Overriding |  |
|  | Javascript supports overriding not overloading. When you define multiple functions with same name, the last one defined will override all the previously defined ones.  function multiplyNumber(x, y, z) {  return x \* y \* z  }  function multiplyNumber(x, y) {  return x \* y  }  multiplyNumber(2, 3, 4) // 6  We can also override javascript built-in functions |  |
| 9 | Interfaces |  |
| 10 | Composition |  |
|  |  |  |
| 3 | **Scope & Hoisting** | |
| 1 | Nested Scope |  |
| 2 | Lexical Scope |  |
| 3 | Dynamic Scope |  |
| 4 | Scope Chain |  |
| 5 | Lexical Scope |  |
| 6 | Hoisting |  |
|  |  |  |
| 4 | **Context** |  |
| 1 | Context |  |
| 2 | this keyword |  |
| 3 | call, bind and apply functions |  |
|  |  |  |
| 6 | Events |  |
| 1 | Event Delegation/bubbling/propagation |  |
| 2 | Event Capturing |  |
| 3 | Event Listening |  |
| 4 | Window Events |  |
|  |  |  |
| 7 | Functions |  |
| 1 | Function Declaration |  |
| 2 | Function expression |  |
| 3 | Anonymous Function |  |
| 4 | Black Box Functions |  |
| 5 | Private Functions |  |
| 6 | Factory Function |  |
| 7 | Self invoking Functions |  |
| 8 | Iterator Functions |  |
| 9 | IIFE |  |
| 10 | Closure |  |
| 11 | Pure Functions |  |
| 12 | First Class Functions |  |
| 13 | Higher Order Functions |  |
| 14 | Static methods in functions |  |
| 15 | Mixin |  |
|  |  |  |
| 8 | Array |  |
| 1 | Typed Array |  |
| 2 | Array Methods |  |
|  |  |  |
| 9 | **Object Oriented Concepts** |  |
| 1 | Abstractions |  |
| 2 | Encapsulation |  |
| 3 | Polymorphism |  |
| 4 | Inheritance |  |
|  |  |  |
| 10 | **Design Patterns** |  |
| 1 | Creational Design Patterns |  |
| 2 | Factory Pattern |  |
| 3 | Constructor Pattern |  |
| 4 | Singleton Pattern |  |
| 5 | Prototype Pattern |  |
| 6 | Structural Design Pattern |  |
| 7 | Decorator Pattern |  |
| 8 | Composite Pattern |  |
| 9 | Façade Pattern |  |
| 10 | Proxy Pattern |  |
| 11 | Behavioral Pattern |  |
| 12 | Iterator |  |
| 13 | Observer |  |
| 14 | Chain of Responsibility |  |
| 15 | Module Pattern |  |
| 16 | Revealing Module Pattern |  |
| 17 | Mixin Pattern |  |
| 18 | FlyWeight Pattern |  |
|  |  |  |
| 11 | **ES-6** |  |
| 1 | Arrow Functions |  |
| 2 | De-structuring |  |
| 3 | Classes |  |
| 4 | Modules |  |
| 5 | Template Literal |  |
| 6 | Let and const |  |
| 7 | Spread Operator |  |
| 8 | Rest parameter |  |
| 9 | Default Parameter |  |
| 10 | Symbols |  |
| 11 | Generators |  |
| 12 | Async await |  |
| 13 | Promises |  |
| 14 | Getter and Setter |  |
| 15 | Object Literal |  |
| 16 | Set and weakSet |  |
| 17 | Map and weakMap |  |
|  |  |  |
| 12 | Operators |  |
| 1 | Double & triple equal |  |
| 2 | instance of operator |  |
| 3 | try catch |  |
| 4 | throw statement |  |
| 5 | finally statement |  |
| 6 | console object |  |
| 7 | Escaping Character |  |
| 8 | new keyword |  |
| 9 | typeof |  |
| 10 | Undefined vs Undeclared |  |
|  |  |  |
| 13 | **Functional Javascript** |  |
| 1 | Side Effects |  |
| 2 | Currying |  |
| 3 | Recursion |  |
| 4 | Observables |  |
| 5 | Functors |  |
| 6 | Monalds |  |
| 7 | Transducers |  |
| 8 | Lenses |  |
| 9 | enum |  |
| 10 | Data normalization |  |
|  |  |  |
| 14 | **Terms/Concepts** |  |
| 1 | Reactive Programming |  |
| 2 | The Temporal Dead Zone |  |
| 3 | Shallow copy |  |
| 4 | Strict Mode |  |
|  |  |  |
| WEB PERFORMANCE | | |
|  | Async/Defer |  |
|  | Prefetch |  |
|  |  |  |
| WEB SECURITY | | |
|  | XSS – Cross site scripting |  |
|  | CORS |  |
|  | CSRF – Cross Site Request Forgery |  |
|  | Vulnerability |  |
|  |  |  |
| STORAGE | | |
|  | Cookie |  |
|  | Session Storage |  |
|  | Local Storage |  |
|  | Date() |  |
|  |  |  |
|  |  |  |
| ARCHITECTURE | | |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
| Functions | | |
|  | setTimeout() |  |
|  | clearTimeout() |  |
|  | setInterval() |  |
|  | addEventListner() |  |
|  | .call() |  |
|  | .apply() |  |
|  | .bind() |  |
|  | Object.assign() |  |
|  | Object.create() |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |
| --- | --- |
| INTERVIEW QUESTIONS | |
| 1 | Sort an array of numeric numbers |
| 2 | Sort an array in reverse order |
| 3 | Sort an array ignoring small and caps character |
| 4 | Whats difference in objects created from Object.create() and ‘new’ |
| 5 | Get the lowest value in an array |
|  | array.sort( (a,b) => a – b)[0] |
| 6 | Get the highest value in an array |
|  | array.sort( (a,b) => b - a)[0] |
| 7 | sorting an array in descending alphabatical order |
|  | array.sort().reverse() |

Difference in constructor and normal function

Constructors name should start capital letter

How to set private properties in constructor

if new instance of constructor is called withour using new keyword, it will return undefined because constructor is not returning anything

how to avoid users creating instance without using new keyword