QUIZ - 41

(1.) What is the purpose of using try-catch block in programming?

Ans. **To handle exceptions that occur during program execution**

To prevent errors from occurring in the program

To terminate the program in case of an error

To optimize the performance of the program

(2.) In which section of a try-catch block is the code that may throw an exception placed?

Ans. **try**

catch

finally

both try and catch

(3.) Which of the following statements is true about try-catch blocks?

Ans. A try block must always be followed by a catch block

A catch block must always be followed by a finally block

A finally block is optional

A try block can only contain one catch block

(4.) Which of the following is the correct syntax for a try-catch block in Javascript?

Ans.

```
try {
        // code that may throw an exception
} catch (Exception e) {
        // code to handle the exception
}
```

```
try (
        // code that may throw an exception
) catch {
        // code to handle the exception
}
```

```
try {
        // code that may throw an exception
} finally {
        // code to execute regardless of whether an exception occurs or not
}
```

```
try {
        // code that may throw an exception
} except {
        // code to handle the exception
}
```

(5.) What is the purpose of using getters and setters in JavaScript?

Ans. **To access and modify private properties of an object**

To create new properties in an object

To delete existing properties from an object

None of the above

(6.) Which of the following is the correct syntax for a getter in JavaScript?

Ans. **get myProperty() { return this.myProperty; }**

set myProperty() { return this.myProperty; }

get myProperty() { this.myProperty; }

set myProperty() { this.myProperty; }

(7.) Which of the following is the correct syntax for a setter in JavaScript?

Ans. get myProperty() { return this.myProperty; }

set myProperty(value) { this.myProperty = value; }

myProperty() { return this.myProperty; }

myProperty(value) { this.myProperty = value; }

(8.) Which of the following is an advantage of using getters and setters in JavaScript?

Ans. **They allow for encapsulation and data hiding**

They improve performance of the program

They allow for more flexible syntax in the program

They prevent errors from occurring in the program

(9.)Which keyword is used to create a constructor function in a class in JavaScript?

Ans. **constructor**

class

this

new

(10.) Which of the following is the correct syntax for creating a parameterized constructor in a class in JavaScript?

Ans. constructor { }

constructor()

constructor(args) { }

constructor(args = []) { }

(11.) Which of the following is the correct way to define a method inside a class in JavaScript?

Ans. **myMethod() { }**

function myMethod() { }

class myMethod() { }

this.myMethod() { }

(12.) Which of the following is the correct way to create a static method in a class in JavaScript?

Ans. **static myMethod() { }**

function static myMethod() { }

class myMethod() { static }

this.myMethod() { static }

(13.) Which of the following is the correct way to access a static method of a class in JavaScript?

Ans. **MyClass.myMethod();**

obj.myMethod();

this.myMethod();

new MyClass().myMethod();

(14.) What is a prototype in JavaScript?

Ans. It is a function that creates new objects

It is an object that is used as a template for creating other objects

It is a method for accessing private properties of an object

None of the above

(15.) Which of the following is the correct syntax for accessing the prototype of an object in JavaScript?

Ans. obj.prototype

obj.__proto__

obj.getPrototype()

obj.setPrototype()

(16.) Which of the following is the correct way to create a new object using a prototype in JavaScript?

Ans. var newObj = Object.prototype;

var newObj = Object.create(myPrototype);

var newObj = myPrototype.new();

var newObj = myPrototype.create();

(17.) Which of the following is an advantage of using prototypes in JavaScript?

Ans. **They allow for easy inheritance and sharing of properties and methods among objects**

They improve performance of the program

They allow for more flexible syntax in the program

They prevent errors from occurring in the program

(18.) Which of the following is the correct way to add a new method to the prototype of an object in JavaScript?

Ans. obj.prototype.newMethod = function() { }

obj.__proto__.newMethod = function() { }

obj.getPrototype().newMethod = function() { }

obj.setPrototype().newMethod = function() { }

(19.) What is a closure in JavaScript?

Ans. It is a function that returns another function

It is a way of hiding variables and functions from the global scope

It is an inner function that has access to the outer function's variables and parameters

None of the above

(20.) Which of the following is an example of a closure in JavaScript?

Ans. function add(x, y) { return x + y; }

function outer() { var x = 10; return function inner() { return x; } }

function printName(name) { console.log("Hello " + name); }

function multiply(x, y) { return x * y; }

(21.) Which of the following is the correct way to create a closure in JavaScript?

Ans. **Declare an inner function inside an outer function and return it**

Declare an inner function inside an outer function without returning it

Declare a function outside of any other function

Declare a function with the same name as another function

(22.) Which of the following is an advantage of using closures in JavaScript?

Ans. **They allow for encapsulation and data hiding**

They improve performance of the program

They allow for more flexible syntax in the program

They prevent errors from occurring in the program

(23.) Which of the following is an example of using closures for private data in JavaScript?

Ans. function add(x, y) { return x + y; }

function counter() { var count = 0; return function() { count++; return count; } }

function printName(name) { console.log("Hello " + name); }

function multiply(x, y) { return x * y; }

(24.) Which of the following is the correct way to access a getter or setter in JavaScript?

Ans. **obj.myProperty;**

obj.get.myProperty();

obj.get.myProperty;

obj.set.myProperty();

(25.) What is a common use case for closures in JavaScript?

Ans. **Creating private variables and functions.**

Accessing global variables from within functions.

Creating new objects based on existing objects.

Executing functions in parallel.