Mock Test 2 - Web Development - PPT - PW Skills

Section 1: MCQs
Section 2: Coding Questions

* Indicates required question

Email *

shivam.jha.2712@gmail.com

Name *

Shivam Jha

Phone Number *

7643938157

1. Which operator returns true if the two compared values are not equal? * 2 points

- <>
- ~
- ==!
- !==

2. How is a forEach statement different from a for statement? * 2 points

- Only a for statement uses a callback function.
- A for statement is generic, but a forEach statement can be used only with an array.
- Only a forEach statement lets you specify your own iterator.
- A forEach statement is generic, but a for statement can be used only with an array.

3. How does a function create a closure? * 2 points

- It reloads the document whenever the value changes.
- It returns a reference to a variable in its parent scope.
- It completes execution without returning.
- It copies a local variable to the global scope.

4. Which property references the DOM object that dispatched an event? * 2 points

- [ ] self
- [ ] object
- [x] target
- [ ] source

5. Which document method is not used to get a reference to a DOM node? * 2 points

- [x] document.getNode();
- [ ] document.getElementsByClassName();
- [ ] document.querySelectorAll();
- [ ] document.querySelector();

6. What are the states of a promise? * 2 points

- [ ] Pending
- [ ] Fulfilled
- [ ] Rejected
- [x] All of the above

7. Which of the following statements is true about promises? * 2 points

- ○ Promises can only represent successful completion of an asynchronous operation.
- ◉ Promises can represent both successful and unsuccessful completion of an asynchronous operation.
- ○ Promises can only represent unsuccessful completion of an asynchronous operation.
- ○ Promises cannot be used for asynchronous operations.

8. What is the purpose of the "resolve" function in a promise? * 2 points

- ○ To handle errors that occur during an asynchronous operation.
- ◉ To indicate successful completion of an asynchronous operation and provide the result value.
- ○ To cancel an ongoing asynchronous operation.
- ○ To reject the promise and provide an error value.

9. Which of the following methods is used to handle the successful result of a promise? * 2 points

- ○ catch()
- ○ finally()
- ◉ then()
- ○ resolve()

10. Which of the following methods is used to handle errors in a promise? * 2 points

- ○ reject()
- ○ fail()
- ◉ catch()
- ○ error()

11. Which of the following methods can be used to create a new promise in JavaScript? * 2 points

- ○ new Promise()
- ○ Promise.resolve()
- ○ Promise.reject()
- ◉ All of the above

12. What does the "catch()" method do in a promise chain? * 2 points

- ○ Handles the successful result of the promise.
- ◉ Adds a callback to be executed when the promise is rejected.
- ○ Cancels the promise chain.
- ○ None of the above.

13. What is the purpose of the "finally()" method in a promise chain? * 2 points

- ○ To handle errors that occur during the promise chain.
- ○ To indicate successful completion of the promise chain.
- ◉ To add a callback that will be executed regardless of the promise chain's outcome.
- ○ To cancel the promise chain.

14. What does the "Promise.all()" method do? * 2 points

- ○ Executes multiple promises in parallel and returns an array of their results.
- ○ Executes multiple promises sequentially and returns the first resolved promise.
- ○ Executes multiple promises sequentially and returns an array of their results.
- ○ Executes multiple promises in parallel and returns the first resolved promise.

15.Which of the following is a valid way to handle multiple promises simultaneously? * 2 points

- ○ Using nested callbacks
- ○ Using multiple try-catch blocks
- ○ Using Promise.all()
- ○ Using synchronous loops

16. What happens if a promise is resolved with another promise? * 2 points

- The resolved promise is ignored.
- The resolved promise is rejected.
- The resolved promise is chained and its result is used.
- The resolved promise causes an error.

17. What's the output? * 2 points

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

- Lydia and undefined
- Lydia and ReferenceError
- ReferenceError and 21
- undefined and ReferenceError

18. Can a promise be resolved or rejected multiple times? * 2 points

- Yes, but only in exceptional cases.
- Yes, any number of times.
- No, once resolved or rejected, a promise's state cannot be changed.
- No, a promise can only be resolved, not rejected.

19. Which of the following is an advantage of using promises over callbacks? * 2 points

- Promises provide better performance in asynchronous operations.
- Promises eliminate the need for error handling in asynchronous operations.
- Promises make asynchronous code easier to read and maintain.
- Promises are not widely supported in modern JavaScript environments.

20. When are closures commonly used in JavaScript? * 2 points

- When working with asynchronous code and callbacks.
- When defining classes and object-oriented programming.
- When implementing mathematical algorithms and computations.
- When handling user interactions and events in the browser.

21. What's the output? * 2 points

```
let c = { greet: 'Hi!' };

let d;


d = c;

c.greeting = 'Hello!';

console.log(d.greeting);
```

- Hello!
- Hi!
- undefined
- None of the above

22. Which of the following best describes the concept of "lexical scope"? * 2 points

- The scope determined by the order of function calls at runtime.
- The scope is determined by the physical location of the code in the file.
- The scope determined by the order of variable declarations in the code.
- The scope determined by the location of variable references in the code.

23. How can closures help with encapsulation in JavaScript? * 2 points

- By providing a mechanism for creating private variables and functions.
- By allowing variables to be accessed from any scope within the program.
- By automatically handling memory management and garbage collection.
- By enabling seamless integration with external libraries and frameworks.

24. How does a closure retain access to its lexical scope variables even after the outer function has finished executing? * 2 points

- By storing the variables in the global scope
- By keeping a reference to the variables in memory
- By using the "var" keyword to declare the variables
- By automatically inheriting the variables from the parent scope

25. All objects have prototypes. * 2 points

- True
- False

Next Clear form