Technical Questions

**Q1. What are the datatypes available in Javascript?**

**Ans.** Number, Boolean, String, Null, Undefined, Symbol, Array and Object

**Q2**. **Explain == Vs. === operator.**

**Ans** == -> Equality Check example 1 == ‘1’. === -> Type and Equality Check

**Q3. What are closures?**

**Ans.** Functions that are assigned to a variable. They can be passed as an argument in another function.

**Q4. What is the result of the expression console.log(1 + ‘1’) ?**

**Ans.** 11

**Q5 What are higher order functions? Give one practical example.**

**Ans.** Function that accepts other functions as an argument. Example Promise

**Q6 What is the purpose of the bind() method in Javascript?**

**Ans.** Bind allows setting the context of a function permanently.

**Q7. What is memoization?**

**Ans.** Technique of storing costly results of costly function calls in a cache for a given input and returning the result for the next function call with the same input.

**Q8. What are Promises in Javascript?**

Ans. Promises are objects used for asynchronous operations.

**Q9. Explain Synchronous and Asynchronous Programming with Example**

**Ans.** In synchronous programming, the program execution occurs sequentially, and each statement blocks the execution until it is completed. In asynchronous programming, multiple tasks can be executed concurrently, and the program doesn’t wait for a task to finish before moving to the next one.

**Q10. What is the difference between using an arrow function (=>) and regular function (function())? Explain in detail.**

**Ans.** Arrow inherits this and the other doesn’t.

**Q11. What is event delegation and why is it useful?**

**Ans**. technique where you attach a single event listener to a parent element to handle events occurring on its child elements. It’s useful for dynamically created elements or when you have a large number of elements.

**Q12. What are generic classes?**

**Ans.** Generic classes encapsulate operations that are not specific to a particular data type. Example. class Enumerable<T> { }

**Q13. What are standalone components in Angular?**

**Ans.** Can be directly imported into Other angular components. Defined by setting standalone=true

@Component({

standalone: true,

selector: 'profile-photo',

})

export class ProfilePhoto { }

@Component({

standalone: true,

imports: [ProfilePhoto],

template: `<profile-photo />`

})

export class UserProfile { }

**Q13. What is a circular dependency and what happens when there’s circular dependency? How can you overcome the issue?**

**Ans.** When 2 modules depend on each other, it creates a circular dependency error. To overcome it, we need to create a 3rd module upon which those two modules depend and have the common code in that 3rd module.

**Q14. How can you accept input in an angular component?**

**Ans**. using @Input() Decorator

**Q15. What is the input transform in a component? Tell me one important attribute of the transform function.**

**Ans.** Function used to change the value of input. The function should be a pure function

**Q16. You are creating a component that can accept child components or native HTML elements. How can you achieve this?**

**Ans.** Use <ng-content>.

**Q17. Explain the select attribute in <ng-conent>**.

**Ans.** The <ng-content select=’card-title’>, for example, will show only the <card-title> component when rendering an element.

**Q18. How can you get a reference to a child component from a parent component?**

**Ans.** Use @ViewChild(ComponentName) component: ComponentName

**Q20. You are creating a dialog box without using any external library in angular. How can you show a component dynamically in angular?**

**Ans.** Using ViewContainerRef

**Q21. Explain Component Vs. Directive in angular?**

Ans. Component -> A functional Element that can be used like native HTML ELement

Directive -> classes that add additional behavior to elements in your Angular applications.

**Q22. Explain some techniques / ways to write code that is flexible?**

**Ans.** Using Singleton Pattern, Dependency Injection

**Q23. What is XSS?**

**Ans.** Running malicious code into a webpage

**Q24.** . **What other security measures can you apply to web applications?**

**Ans.** Avoid XSS Using Html Sanitization, Trusting Safe Url, Using Content-Security-Policy Header, Enforcing Trusted Types, Using CSRF token, Using Proper User Input Validation

**Q25. Explain Binary Search Algorithms.**

**Ans.** Binary search is a search algorithm used to find the position of a target value within a sorted array. It works by repeatedly dividing the search interval in half until the target value is found or the interval is empty. The search interval is halved by comparing the target element with the middle value of the search space.

**Task 1**

**Given a string, return if all occurrences of a given letter are always immediately followed by the other given letter.**

("he headed to the store", "h", "e") ➞ True

# All occurences of "h": ["he", "headed", "the"]

# All occurences of "h" have an "e" after it.

# Return True

('abcdee', 'e', 'e') ➞ False

# For first "e" we can get "ee"

# For second "e" we cannot have "ee"

# Return False

**Task 2**

**Given a string, return if all occurrences of a given letter are always immediately followed by the other given letter.**

The function is given a non-empty balanced parentheses string. Each opening '(' has a corresponding closing ')'.

Compute the total score based on the following rules:

Substring s == () has score 1, so "()" should return 1

Substring s1s2 has total score as score of s1 plus score of s2, so "()()" should return 1 + 1 = 2

Substring (s) has total score as two times score of s, so "(())" should return 2 \* 1 = 2

Return the total score as an integer.