MCQs

Q.1What are block scoped variables in JavaScript?

a.The variables cannot be accessed outside the function in which they are declared.

b.The variables can be used globally.

c.The variable is only accessible inside the block(if-else/for block) inside which it is declared.

d.Both a and c

Ans.The variable is only accessible inside the block(if-else/for block) inside which it is declared.

Q.2JWT stands for ? and it consists of?

a.JSON Web Token, It consists of header, payload, signature

b.JSON Web Token, It consists of header and payload

c.JSON Web Tech, It consists of header, payload, signature

d.JSON Web Tech, It consists of header and payload

Ans.JSON Web Token, It consists of header, payload, signature

Q.3Which of the following is the right way to add a single line comment in JavaScript?

a.#This is a comment

b.##This is a comment

c./This is a comment

d.//This is a comment

Ans.//This is a comment

Q.4Which of the following is the right way to add a multi line comment in JavaScript?

a.#This is a comment#

b.##This is a comment##

c./*This is a comment*/

d.//This is a comment//

Ans./*This is a comment*/

Q.5How to create a Date object in JavaScript?

a.dateObj = new Date([parameters])

b.dateObj = new DateTime([parameters])

c.dateObj = new Date({parameters obj})

d.dateObj = Date([parameters])

Ans.dateObj = new Date([parameters])

Q.6Which of the following is considered a first class citizen in JavaScript?

a.Functions

b.Class

c.Array

d.Object

Ans.Functions

Q.7What will be the output of the following JavaScript code?

```
let a='0';
for(a;a<5;++a)
console.log(a);
```

a.0 1 2 3 4

b.0 1 2 3 4 5

c.5

d.error

Ans.0 1 2 3 4

Q.8What is destructuring in JavaScript?

a.Through destructuring, we can unpack values from arrays or object properties into separate variables.

b.Destructuring allows us to restructure object properties

c.Destructuring allows us to restructure array elements.

d.All of these.

Ans.Through destructuring, we can unpack values from arrays or object properties into separate variables.

Q.9Which statement cannot be used to declare a variable in JavaScript?

a.let

b.var

c.int

d.const

Ans.int

Q.10What will be the output of the below code snippet?

```
let {name, age} = {name: “abc”, age: 1, id: 1}

console.log(name, age);
```

a.abc 1

b.throws error

c.null null

d.undefined undefined

Ans.abc 1

Q.11What will be the output of the below code snippet?

console.log(parseInt("Hello123"), parseInt("123"), parseInt("Hello"), parseInt("123Hello"));

a.error

b.NaN 123 NaN 123

c.123 123 NaN 123

d.NaN 123 NaN NaN

Ans.NaN 123 NaN 123

Q.12What are loops?

a.Repeatedly run a set of statements until a specified condition evaluates to false/ or can even run it infinite times.

b.It does the condition check and runs the specified block of code based on the satisfied condition.

c.Both a and b

d.None of them

Ans.Repeatedly run a set of statements until a specified condition evaluates to false/ or can even run it infinite times.

Q.13What is the worst case time complexity of searching an element in an array using linear search?

a.O(logn)

b.O(n)

c.O(1)

d.O(n^2)

Ans.O(n)

Q.14What is the correct syntax for destructuring arrays in JavaScript?

a.const [a, b, …rest] = [1, 2, 3, 4, 5]

b.const {a, b, …rest} = [1, 2, 3, 4, 5]

c.const a, b, …rest = [1, 2, 3, 4, 5]

d.None of these.

Ans.const [a, b, …rest] = [1, 2, 3, 4, 5]

Q.15Which array method is used to iterate on all the array elements and perform some task/transformation on them and return the new array?

a.map

b.reduce

c.foreach

d.filter

Ans.map

Q.16Which of the following sorting algorithms would give the best time complexity if the given array is already sorted?

a.Insertion Sort

b.Quick Sort

c.Merge Sort

d.All of these

Ans.Insertion Sort

Q.17What will be the output of the below code snippet?

```
let numbers = [1,2,3,4,5,6,7,8];

numbers.splice(2, 4, 4)

console.log(numbers.filter(x => x%2 == 0));
```

a.[2, 4, 6]

b.[2, 4, 8]

c.[1, 3, 5, 7] d.[2, 4, 6, 8]

Ans.[2, 4, 8]

Q.18What is the time, space complexity of the following code?

```
let a = 1, b = 0;
for (i = 0; i < N; i++) {
        for (j = 0; j < M; j++) {
                let temp = b;
b = a + b;
a = temp;
}
}
```

a.O(N * M) time, O(1) space b.O(N + M) time, O(N + M) space

c.O(N + M) time, O(1) space d.O(N * M) time, O(N + M) space

Ans.O(N * M) time, O(1) space

Q.19What is the worst case time complexity of binary search?

a.O(logn) b.O(n)

c.O(1) d.O(n^2)

Ans.O(logn)

Q.20Will the following JavaScript code work?

```
var tensquared = (function(x) {return x*x;}(10));
```

a.Yes, perfectly

b.Error

c.Exception will be thrown

d.Memory leak

Ans.Yes, perfectly

Q.21What will be the output of the below code snippet?

```
        function multiplyX(x){
                return function(y){
                        return x * y;
}
}
console.log(multiplyX(5)(5))
```

a.25

b.0

c.10

d.error

Ans.25

Q.22What will be the output of the following JavaScript code?

```
        console.log([2]==2)
        console.log([2]===2)
```

a.false true

b.true false

c.false false d.true true

Ans.true false

Q.23Bubble sort has the worst case time complexity of?

a.O(n^2) b.O(nlogn)

c.O(logn) d.O(n)

Ans.O(n^2)

Q.24The below code depicts which concept of JavaScript through which internal function is able to access num1 which is the variable of the outer function?

```
    function addX(num1){
    return function (num2){
    return num1 + num2;
}
}
console.log(addX(5)(5)) //10
```

a.Hoisting b.Closure

c.Both a and b d.None of these

Ans.Closure

Q.25What will be output for the following code?

```
function demo()
{
        console.log("Hello");
        demo();
        return 0;
}
```

a.Hello will be printed only once

b.Hello infinite number of times and stack overflow error will be thrown

c.Hello is not printed at all

d.0 is returned

Ans.Hello infinite number of times and stack overflow error will be thrown

Q.26What is the time, space complexity of the following code?

```
let a = 1, b = 0;
for (i = 0; i < N; i++) {
        a = a * a
}
for (j = 0; j < M; j++) {
        b = b + b;
}
```

a.O(N * M) time, O(1) space

b.O(N + M) time, O(N + M) space

c.O(N + M) time, O(1) space

d.O(N * M) time, O(N + M) space

Ans.O(N + M) time, O(1) space

Q.27Which of the following is not an inplace sorting algorithm?

a.Insertion Sort

b.Bubble Sort

c.Quick Sort

d.Merge Sort

Ans.Merge Sort

Q.28What is the time complexity of the following code?

```
        let i, j, k = 0;
for (i = 1; i <= n; i = i * 2) {
        for (j = n / 2; j <= n; j++) {
                k = k + n / 2;
        }
}
```

a.O(n)

b.O(nLogn)

c.O(n^2)

d.O(n^2Logn)

Ans.O(nLogn)

Q.29What is the worst case time complexity of the selection sort?

a.O(n)

b.O(nLogn)

c.O(n^2)

d.O(n^2Logn)

Ans.O(n^2)

Q.30What is the correct syntax for creating a new html element and adding it before an element with id `list`?

a.const newDiv = document.createElement("div");

document.body.appendBefore('list', newDiv);

b.const newDiv = document.createElement();

document.addChild(newDiv);

c.const newDiv = document.createElement("div");

var listElement = document.getElementById("list");

document.body.insertBefore(newDiv, listElement);

d.const newDiv = document.createElement("div");

var listElement = document.getElementById("list");

document.body.appendBefore(newDiv, listElement);

Ans.const newDiv = document.createElement("div");

var listElement = document.getElementById("list");

document.body.insertBefore(newDiv, listElement);

Q.31What is the worst case time complexity of selection sort?

a.O(n^3)

b.O(nlogn)

c.O(n^2)

d.O(n)

Ans.O(n^2)

Q.32The popular notion to describe stack is?

a.Last in First out

b.Frist in First out

c.None of the above

d.Both a and b

Ans.Last in First out

Q.33What is the best case time complexity of selection sort?

a.O(n^3)

b.O(nlogn)

c.O(n^2)

d.O(n)

Ans.O(n)

Q.34In which sorting algorithm, we use frequency of the elements to sort the array?

a.Bucket Sort

b.Bubble Sort

c.Count Sort

d.Merge Sort

Ans.Count Sort

Q.35What is the time complexity of finding an element in a Singly LinkedList?

a.O(nlogn)

b.O(n)

c.O(1)

d.O(logn)

Ans.O(n)

Q.36Which of the following is not a stable sorting algorithm?

a.Quick Sort

b.Bubble Sort

c.Count Sort

d.Merge Sort

Ans.Quick Sort

Q.37When we try to remove the element from the stack, and if the stack is already empty, this situation can be

described as?

a.Underflow

b.Overflow

c.None of the above

d.Both a and b

Ans.Underflow

Q.38In queue, we push the element at _____ position

a.top

b.end

c.middle

d.None of the above

Ans.top

Q.39Which of the following statement is true about LinkedList?

a.LinkedList is a non-linear data structure

b.Elements in linkedlist are stored in contiguous memory location

c.Linkedlist can not shrink during program execution

d.None ofthe above

Ans.None of the above

Q.40In which sorting algorithm, we sort the array with the least significant digit then move towards the most significant digit

a.Quick Sort

b.Bubble Sort

c.Count Sort

d.Radix Sort

Ans.Radix Sort

Round 2 DSA Problems

Problem1. Admission

Problem Statement

You have to implement a program to find the eligibility of admission for Engineering based on following criteria
Marks Scored in Maths should be greater then or equal to 65 (>=65)
In Physics, marks should be atleast equal to or greater than 55 (>=55)
And for Chemistry it should be greater then or equal to 50 (>=50) and either one of the below 2 conditions should be true:
Sum of the marks scored in all three subjects should be >= 195 or Sum of marks scored in Math + Physics >= 140

Constraint

• 0 <= Score <= 100 for each subject

Input Format

• Space separated 3 integers denoting Math, Physics and Chemistry score
Eg. 65 71 52 -> Math 65, Physics 71, Chemistry 52

Output Format

• Print “Eligible” if eligibility criteria is fulfilled else print “Not eligible”

Sample Input 1

65 71 52

Sample Output 1

Not eligible

Explanation of Sample 1

Total Score -> 65 + 71 + 52 = 188 < 195 and Maths and physics score sum -> 65 + 71 = 136 < 140. Since criteria is not fulfilled, Not eligible

Sample Input 2

70 75 52

Sample Output 2

Eligible

Explanation of Sample 2

Individual subject score criteria is fulfilled as well as total Score -> 70 + 75 + 52 = 197 > 195 hence Eligible

Sample Input 3

55 80 90

Sample Output 3

Not eligible

Explanation of Sample 3

Since Maths Score is < 65

Sample Input 4
70 71 52
Sample Output 4
Eligible
Explanation of Sample 4
Total Score -> 70 + 71 + 50 = 191 < 195 and Maths and physics score sum -> 70 + 71 = 141 > 140. Though total score is > 190 but individual subject score criteria is fulfilled as well as Math and physics score sum > 140 hence Eligible

Template:

```
let arr = readline().split(" ").map(x => parseInt(x)).slice(0, 3);
//maths = arr[0], physics = arr[1], chemistry = arr[2]
function admissionEligibility(arr)
{
   //write your logic here. Return the output
}

console.log(admissionEligibility(arr));
```

Solution:

```
let arr = readline().split(" ").map(x => parseInt(x)).slice(0, 3);
//maths = arr[0], physics = arr[1], chemistry = arr[2]
function admissionEligibility(arr){
   let [m, p, c] = arr;
   if(m < 65 || p < 55 || c < 50){
      return "Not eligible";
   } else if((m + p + c) < 195 && (m + p) < 140 ) {
      return "Not eligible";
   } else {
      return "Eligible" ;
   }
   return 0;
}
console.log(admissionEligibility(arr));
```

2. Standing Rows

Problem Statement

You are given an integer array height, where each element of an array represents a student's height. Students were asked to stand in a non-decreasing order (small to large) on the basis of their height. The given array represents the current position of each student. You have to figure out how many students are not at their expected position i.e

heights[i] != expected[i], where expected represents the correct position

Constraint

• 1 ≤ n ≤ 100
• 1 ≤ heights[i] ≤ 100

Input Format

• The first line of input consists of a single integer N denoting the number of students.
• The second line of input consists of n space separated integers height[0], height[1], ..., height[n], where height[i] denotes the height of the student.

Output Format

•Return the number of students whose heights[i] != expected[i]

Sample Input 1

6
1 1 4 2 1 3

Sample Output 1

3

Explanation of Sample 1

heights: [1,1,**4**,2,**1**,**3**]
expected: [1,1,**1**,2,**3**,**4**]
Indices 2, 4, and 5 do not match

Sample Input 2

5
5 1 2 3 4

Sample Output 2

5

Explanation of Sample 2

heights: [5,1,2,3,4]
expected: [1,2,3,4,5]
All indices do not match

Sample Input 3

5
1 2 3 4 5

Sample Output 3

0

Explanation of Sample 3

heights: [1,2,3,4,5]

expected: [1,2,3,4,5]

All indices match

Solution:

```
let n = parseInt(readline());
let heights = readline().split(" ").map(x => parseInt(x)).slice(0, n);

function standingRows(heights){
   let expectedHeights = heights.map(x => x).sort((a,b) => a-b);
   let count = 0;
   for(let i = 0; i < heights.length; i++){
      if(heights[i] != expectedHeights[i]) count++;
   }
   return count;
}
console.log(standingRows(heights));
```

Template:

```
let n = parseInt(readline());
let heights = readline().split(" ").map(x => parseInt(x)).slice(0, n);

function standingRows(heights){
   //write your logic here
}

console.log(standingRows(heights));
```

Feature:

Round1: App building

Build a simple URL shortener. You might have used a URL shortener before, such as bit.ly, goo.gl. They are useful for shortening long URLs so that you can easily share them with your friends, family or co-workers.

How does URL shortener work?
When the user inputs the url, suppose www.relevel.com, the shortener system passes the long url through the hash function, which returns a short id, which is then used to map it to the entered url.
For eg.

User enters www.relevel.com -> hash function returns -> Uy8fu
So when the request is made for ourdomain.com/Uy8fu it is redirected to www.relevel.com

Shortened version of www.relevel.com would be localhost:3000/Uy8fu. When you hit this url it should redirect you to www.relevel.com

For redirection. You can use res.redirect(url) if you are using express lib

Hash function basically takes a variable size input and maps it to fixed sized values which we call hashcode.

Step1: User enters a url
Step2: Pass the url through the hash function to generate hash code
Step3: Store the url and shortened url in db and return the shortened url

Hash function is provided below which you can use to build the url shortener:

```
function getHash() {
        var text = '';
        var possible = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';
        for (var i = 0; i < 5; i++)
                text += possible.charAt(Math.floor(Math.random() * possible.length));
        return text;
}
```

Apis should be as follows
GET /?url=www.relevel.com -> returns the shortened url
GET /allUrls -> returns the list of all urls along with the shortened version

Boilerplate code is available :- https://github.com/VJ28/boilerplate_code

Round2: Enhancement

Add login into your system, and the user should be able to see their shortened url. If user1 shortens a few urls, they should be able to view the list of urls and their shortened version.

You will need to add userId along with url and shortened url in db

GET /shortUrls -> returns the list of all urls along with the shortened version for that user
POST /signup
POST /login

Round3: Optimization

When we talk about hash function, there is a concept of collision i.e. a hash function can return the same value for two different urls, which could break our system flow. So you have to handle that so that our system should not return the same shortened url for two different urls.
To overcome this, large systems keep a cache layer(it is an in-memory object due to which we can access it very fast, we only store some MBs in cache) to perform a look up if the generated hash code is already present or not, as looking up in cache is very efficient. But considering the scope of this test, you can basically lookup in the database if the generated hash is already present or not, if not it is safe to use or else generate different hash

Step1: User enters a url
Step2: Pass the url through the hash function to generate hash code
Step3: Check if the generated hashcode is already present in the database Step4: If present, repeat step2
Step5: Else store the hash and return the shortened url

Suppose www.relevel.com shortened the version to localhost:3000/Uy8fu. When you hit this url it should redirect you to www.relevel.com

For redirection. You can use res.redirect(url) if you are using express lib

Solution Links : https://github.com/subhahuRelevel/13thMarchTestSolution