JAVASCRIPT DAY 17

VARIABLES PRACTICE PAPER QUESTIONS AND ANSWER

1) Why variables?

Variables are used to store data values in a program. These values can be updated and reused throughout the program, making the code dynamic and efficient.

2) How to declare variables?

Variables in JavaScript can be declared using:

1. **var: Function-scoped, can be redeclared and reassigned.**
2. **let: Block-scoped, cannot be redeclared but can be reassigned.**
3. **const: Block-scoped, cannot be redeclared or reassigned.**
4. **No keyword: Creates a global variable (not recommended).**

Example:

```
var a = 10;
let b = 20;
const c = 30;
d = 40; // No keyword
```

3) Rules to declare variables:

1. **Variable names must start with a letter, $, or _.**
2. **Cannot start with a number.**
3. **Variable names are case-sensitive.**
4. **Should not use reserved keywords (e.g., let, class, return).**

4) What is a datatype?

A datatype represents the type of data a variable holds.

5) How many types of datatypes?

JavaScript has two types:

1. **Primitive datatypes (7 types): string, number, boolean, undefined, null, bigint, symbol.**
2. **Non-primitive datatypes: object, including arrays, classes, and interfaces.**

6) Differences Between Primitive and Non-Primitive Datatypes:

Primitive	**Non-Primitive**
Stores single, immutable data	**Stores collections or objects**
Example: string, number	**Example: array, object**
Immutable (cannot change)	**Mutable (can change)**

7) Primitive Datatypes:

1. **String**
2. **Number**
3. **Boolean**

4. **Undefined**
5. **Null**
6. **Bigint**
7. Symbol

8) Non-Primitive Datatypes:

1. **Object**
2. **Array**
3. **Class**
4. **Interface**

9) JavaScript Program: Addition

```
let x = 100;
let y = 200;
let z = x + y;
console.log(z); // Output: 300
```

10) JavaScript Program: Square

```
let x = 10;
let y = x ** 2; // or x * x
console.log(y); // Output: 100
```

11) What is a string?

A string is a sequence of characters enclosed in single ('), double ("), or backtick (`) quotes.

12) How to declare a string?

```
let str1 = 'Hello';
let str2 = "World";
let str3 = `Welcome`;
```

13) Result:

```
var firstname = "ExcelR";
var lastname = "EduTech";
var fullname = firstname + " " + lastname;
document.write(fullname);
```

Output:

```
ExcelR EduTech
```

14) Result:

```
var sub = `fullstack`;
var msg = `welcome to ${sub}`;
document.write(msg);
```

Output:

welcome to fullstack

15) Points on Backtick Operator (Template Literal):

- **Introduced in ES6.**
- **Supports variable interpolation using ${}.**
- **Allows multi-line strings.**
- **Simplifies string concatenation.**

16) Boolean Values:

1. **true**
2. **false**

17) Ternary Operator Syntax:

condition ? expressionIfTrue : expressionIfFalse;

18 & 19) Results:

1. **9 > 8 > 7 outputs "dotnet". Explanation: (9 > 8) is true, true > 7 is false.**
2. **1 < 2 < 3 outputs "java". Explanation: (1 < 2) is true, true < 3 is true.**

20) NaN:

NaN stands for "Not a Number".

21) Differences Between Undefined and Null:

Undefined	**Null**
Value not assigned to variable.	**Represents no value.**
No initialization required.	**Explicit initialization required.**
Arithmetic operations result in NaN.	**Treated as 0 in arithmetic operations.**

22) Differences Between == and ===:

== (Equality)	**=== (Strict Equality)**
Compares values after type conversion.	**Compares value and type.**
Example: 5 == "5" → true.	Example: 5 === "5" → false.

23) let and const Version:

Introduced in ES6.

24 & 25) Bigint Datatype:

- **Range: Beyond 253 - 1.**
- **Introduced in ES11 (2020).**

26) Bigint Suffix:

Append n to a number. Example: let big = 123456789n;.

27) How to Hide Identifiers:

Use closures or let/const within block scope.

28) Differences Between var and let:

Var	Let
Function-scoped.	**Block-scoped.**
Allows redeclaration.	**No redeclaration allowed.**
Hoisted as undefined.	**Hoisted but in Temporal Dead Zone.**

29) Points on const:

- **Block-scoped.**
- **Cannot be reassigned.**
- **Example:**
- **const PI = 3.14**

30) Variable Hoisting:

Variables declared with var are hoisted and initialized as undefined. let and const prevent hoisting issues.

31) Declaring Block:

```
{
  let x = 10;
}
```

32-34) Results for var, let, const:

Redeclaring let or const within the same scope causes errors.

var allows redeclaration.

Example:

```
var x = 100; // OK
let x = 100; // Error if redeclared
const x = 100; // Error if redeclared
```

LOOPS PRACTICE PAPER QUESTIONS

1) How to represent arrays?

In JavaScript, arrays are represented using square brackets []. Example:

```
let arr = [1, 2, 3, 4, 5];
```

2) Index starts from:

The index in JavaScript arrays starts from **0**.

3) How to access array elements?

Array elements can be accessed using their index. Example:

```
let arr = [10, 20, 30];
console.log(arr[0]); // Output: 10
```

4) How to iterate array elements?

You can iterate array elements using loops like for, forEach, for...of, etc. Example:

```
let arr = [1, 2, 3];
for (let i = 0; i < arr.length; i++) {
    console.log(arr[i]);
}
```

5) Write the for loop syntax:

```
for (initialization; condition; increment/decrement) {
    // Code block to execute
}
```

6) Iterate the following array with a for loop:

```
let arr1 = ['Java', 'dotnet', 'ui', 'react', 'angular'];
for (let i = 0; i < arr1.length; i++) {
    console.log(arr1[i]);
}
```

7) Write the forEach() loop syntax:

```
array.forEach(function(element, index, array) {
    // Code block to execute
});
```

8) Iterate the following array with a forEach() loop:

```
let arr1 = ['Java', 'dotnet', 'ui', 'react', 'angular'];
arr1.forEach(function(element) {
    console.log(element);
});
```

9) Write the for...of loop syntax:

```
for (let element of array) {
    // Code block to execute
}
```

10) Iterate the following array with a for...of loop:

```
let arr1 = ['Java', 'dotnet', 'ui', 'react', 'angular'];
for (let element of arr1) {
    console.log(element);
}
```

11) Write the if...else condition syntax:

```
if (condition) {
    // Code block to execute if condition is true
} else {
   // Code block to execute if condition is false
}
```

12) Find the result:

```
if (true) {
   document.write("java");
} else {
   document.write("dotnet");
}
// Output: java
```

13) Write the switch case syntax in JavaScript:

```
switch (expression) {
   case value1:
      // Code block for value1
      break;
   case value2:
      // Code block for value2
      break;
   default:
      // Code block for default
}
```

14) Write one basic example for switch case in JavaScript:

```
let day = 2;
switch (day) {
   case 1:
      console.log("Monday");
      break;
   case 2:
      console.log("Tuesday");
      break;
   case 3:
      console.log("Wednesday");
      break;
   default:
      console.log("Other day");
}
```

15) Find the result:

Given array:

let arr1 = [10, 20, 30, 40, 50];

- arr1[0]: 10
- arr1[4]: 50
- arr1[10]: undefined (out of bounds)
- arr1[-1]: undefined (negative indices are invalid for [])
- arr1.at(0): 10
- arr1.at(-1): 50 (last element)
- arr1.at(-5): 10 (first element)
- arr1.at(-10): undefined (out of bounds)