**TASK (21.09.2021)  
ON-SESSION**

1. **TEMPORAL DEADZONE:**

TDZ is the term that describe a state of variable where they are unreachable even after the hoisting occurred.  
 For example: console.log(food)

var food = ‘pizza’

Output will throw an error called ‘undefined’, since in ‘var’ variable can be accessed and can not access the value of it before it is declared.  
 If we use ‘let’ and ‘const’ in the same program it will throw a error called ‘food is not defined’ because the variable itself can not be accessed before declaring it.  
 The let and constvariables exist in the TDZ from the start of their enclosing scope until they are declared.

1. **SHADOWING IN JS:**

When a variable is declared in a certain scope having the same name defined on its outer scope and when we call the variable from the inner scope, the value assigned to the variable in the inner scope is the value that will be stored in the variable in the memory space. This is known as **Shadowing or Variable Shadowing**.

Example:  
 function func() {

    let a = 'India';

    if (true) {

        let a = 'Asia-Indian';  // New value assigned

        console.log(a);

    }

    console.log(a);

func();

**Output:**

Asia-Indian  
India

# **IIFE -** Immediately invoked function expression:

An Immediately invoked function expression is a way to execute functions as soon as they created.

It is a design pattern which is also known as a [Self-Executing Anonymous Function](https://developer.mozilla.org/en-US/docs/Glossary/Self-Executing_Anonymous_Function) and contains two major parts:

1. The first is the anonymous function with lexical scope enclosed within the Grouping Operator(). This prevents accessing variables within the IIFE idiom as well as polluting the global scope.
2. The second part creates the immediately invoked function expression () through which the JavaScript engine will directly interpret the function.