# JS Assignment [Deadline: 2025-07-30]

**What will this code log? Explain your reasoning.**

*const person = {*

*name: 'Alice',*

*sayHi: function() {*

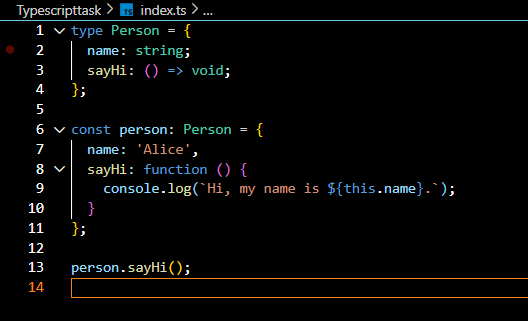
*console.log(`Hi, my name is ${this.name}.`);*

*}*

*};*

*person.sayHi();*

*typescript:*



Output: A black background with white text

AI-generated content may be incorrect.

**What is logged to the console, and why? How would you fix it so it logs "Alice"?**

*const person = {*

*name: 'Alice',*

*greet: function() {*

*console.log(`Hello, ${this.name}`);*

*}*

*};*

*const greetFunction = person.greet;*

*greetFunction();*

*Ans:*

Here according to the code if we log the code as it is it gives the output “hello undefined” because the main problem is in the line “*const greetFunction = person.greet;” here we are just calling or say we are copying the function without its context.so when we call greetfunction() we are just calling a function rather than a method of the person. That’s why its giving a name undefined .*

*Following is the code with fixed solution that logs “Alice”*

*const person = {*

*name: 'Alice',*

*greet: function() {*

*console.log(`Hello, ${this.name}`);*

*}*

*};*

*const greetFunction =()=> person.greet();*

*greetFunction();*

*here the main fix we did was “const greetFunction =()=> person.greet(); “ where we introduce a arrow function rather than a simple function where it doesnot have their own this . here this is used from the surrounding scope when person.greet() is called.*

*Typescript :*

**

*Output: A black background with white text

AI-generated content may be incorrect.*

**What will this code log and why?**

*const user = {*

*name: 'Bob',*

*logName: function() {*

*setTimeout(() => {*

*console.log(this.name);*

*}, 100);*

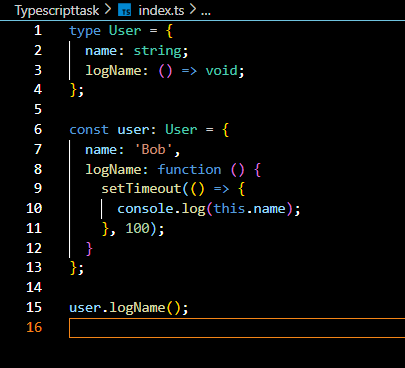
*}*

*};*

*user.logName();*

*Ans: here “Bob” is logged in the console because here this.name is called inside a settimeout function which is a arrow function which doesnot have their own this so they inherit the this property from their surrounding context which is a logname function() which is a method of the user object so basically this.name is same as user.name which refers to “Bob”. Similarly on thr additional note the settimeout function refers to when to log the output . its doesnot effect the console or output it only helps in when to logged in . so according to our code the ouput is logged after 100 ms instead of immediately.*

*Typescript :*

**

*Output:*

**

**You have a User object with a method that logs a welcome message. There's also a "Login" button in your HTML. You want the welcome method to be called when the button is clicked, and the message should use the User object's data.**

*const User = {*

*name: 'Alice',*

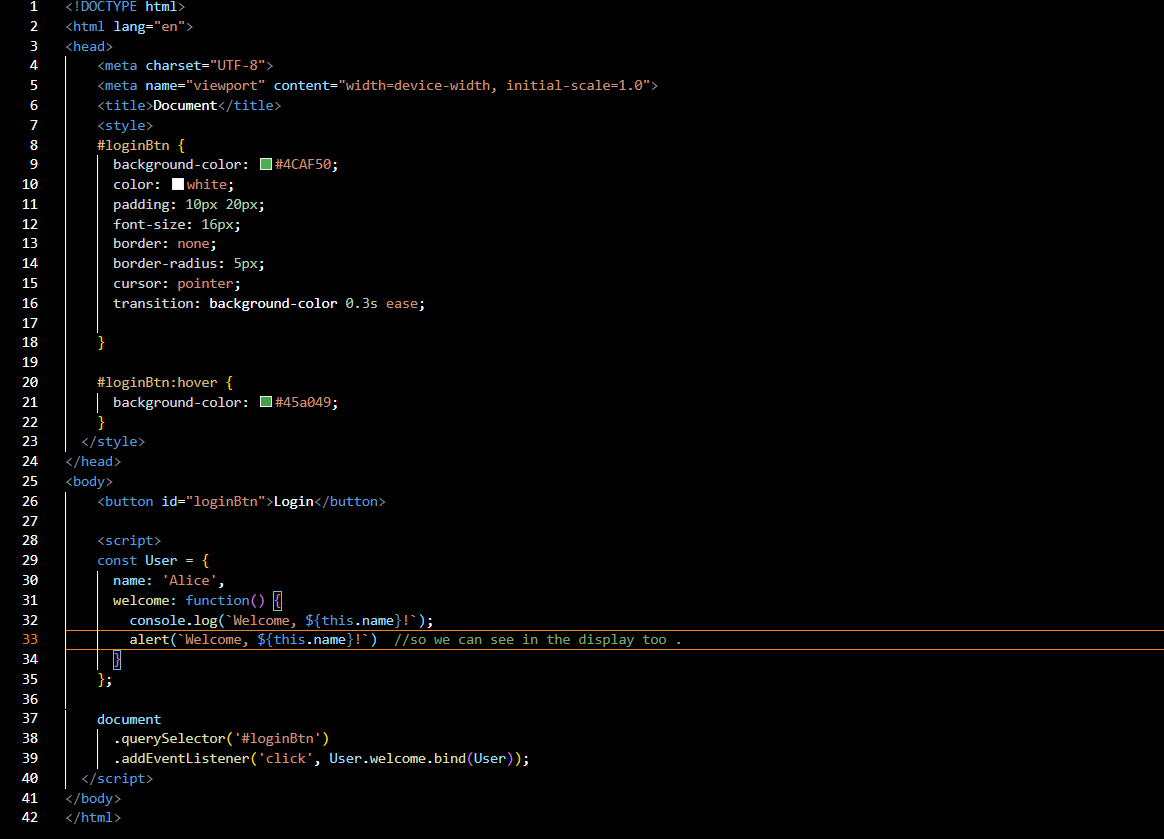
*welcome: function() {*

*console.log(`Welcome, ${this.name}!`);*

*}*

*}*

*Hint: DOM APIs can be used*

**

*Heres my screenshot of the code with button which when clicked console welcome alice also I used alert so we can also see in the display. here what we did is we use .bind(user) which creates a new function where this is set to user . if we just did “document.querySelector('#loginBtn').addEventListener('click', User.welcome);” then here this inside welcome doesnot refer to user anymore . it refers to button element(#loginBtn) which gives this.name as undefined.*

*Screenshot of the output :*

*A screen shot of a computer

AI-generated content may be incorrect.*

**What will be logged to the console? Explain!**

*let animal = 'Cat';*

*function showAnimal() {*

*let animal = 'Dog';*

*console.log(animal);*

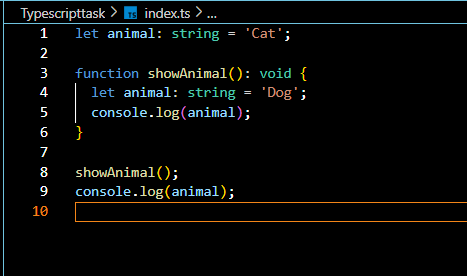
*}*

*showAnimal();*

*console.log(animal);*

*Ans: here the showAnimal() logs Dog as a output and console.log(animal); logs cat as a output because let animal = 'Cat'; is a global scope but let animal = 'Dog'; is a local scope which is with in a function .*

*Typescript :*

**

*Output : *

**What will this code output? Why?**

*function test() {*

*console.log(a);*

*console.log(foo());*

*var a = 1;*

*function foo() {*

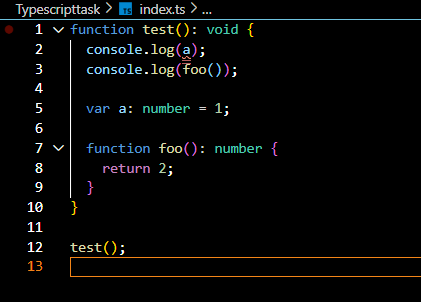
*return 2;*

*}*

*}*

*test();*

*Ans: here the output is undefined and 2 respectively because this code refers to the concept of the hoisting in js . here function declaration arr fully hoisted so that both their name and body are moved to the top of their scope. So the function foo() is available anywhere inside test().but variables cannot be . variables assigned with var can only be hoisted . This means the variable a is known at the top of the function scope, but its value is undefined until the line a = 1; is actually executed.*

*Typescript:* **

Output : A black screen with yellow and white text

AI-generated content may be incorrect.

**Explain the output of this for loop.**

*for (var i = 0; i < 5; i++) {*

*setTimeout(function() {*

*console.log(i);*

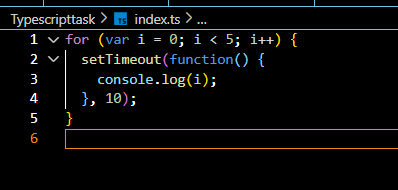
*}, 10);*

*}*

*Ans: here it gives the output 55555 . because as we know var is a function scoped rather than a block scoped because of which All five setTimeout callbacks* ***share the same i****, which is declared once. Here after the completion of the loop I becomes 5 and then all the timeout runs which makes it log the same value i.e 5 .*

*Note : we can fix that with the help of let which is a block scope*

*Typescript :*

**

*A black screen with yellow and green text

AI-generated content may be incorrect.*

**What will the following code log to the console on the last two lines? Explain why the count variable is not reset.**

*function createCounter() {*

*let count = 0;*

*return function() {*

*count++;*

*console.log(count);*

*};*

*}*

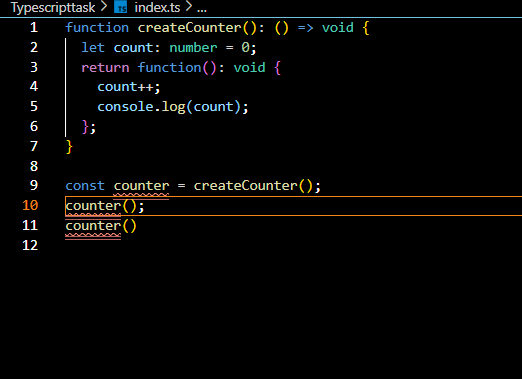
*const counter = createCounter();*

*counter();*

*counter();*

*Ans : the following code line log 1 and 2 respectively on the last two lines . it is using a concept of closure where createCounter() returns a function that has access to the count variable . so even after createcounter has finished executing the inner function remembers the value of count with the help of closure.*

*Typescript:*

**

*Output:*

*A black screen with yellow and white text

AI-generated content may be incorrect.*

**What does this code log? Explain!**

*const myObject = {*

*id: 'my-object',*

*createLogger: function() {*

*return () => {*

*console.log(`Logger for ${this.id}`);*

*};*

*}*

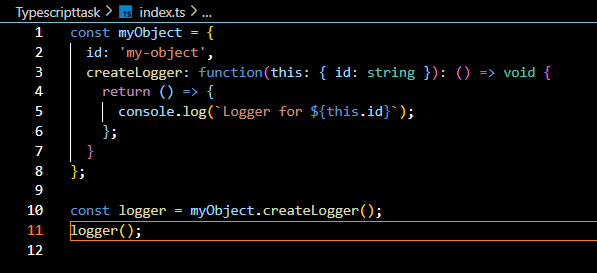
*};*

*const logger = myObject.createLogger();*

*logger();*

Ans: it logs Logger for my-object because here create logger is a method of the object myobject which also have a arrow function inside which doesnot have a this of their own rather it inherits of their surrounding context so this.id refers to myObject.id .

Typescript :





Output:

**Write a function *makeAdder(x)* that takes a number x and returns a new function. The new function should take a number y and return the *sum x + y*. Use a closure to achieve this.**

*Eg:*

*let add = makeAdder(5);*

*console.log(add(2));*

*Answer: 7*

*function makeAdd(x) {*

*return function(y) {*

*return x + y;*

*};*

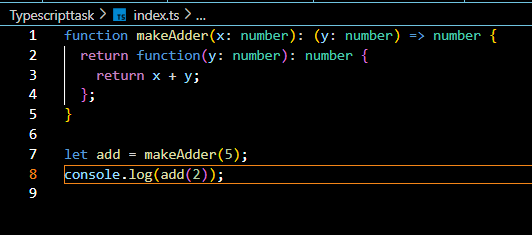
*}*

*let add = makeAdd(5);*

*console.log(add(2));*

*output : 7*

*typescript:*

**

*Output:* **

**Implement a parent Animal and a child Dog relationship in two different ways:**

* Using the Constructor/Prototype pattern.
* Using the ES6 class syntax.

Your solution must satisfy two conditions for both patterns:

* The Animal must have an eat() method that the Dog inherits.
* The Dog must also have its own bark() method.

Prototype pattern typescript :

A screen shot of a computer program

AI-generated content may be incorrect.

Output:

A black screen with white text

AI-generated content may be incorrect.

Es6 pattern :

A screen shot of a computer program

AI-generated content may be incorrect.

Output: 

**Design a content feed/post UI (similar to X/Twitter) involving Post, Comment, and User details.[Using either fetch API or axios and DOM APIs]**

Use the **API** from Resources section: <https://jsonplaceholder.typicode.com/>

**Features:**

* Feed View (like X/Twitter)
* Render a list of posts
* Each post should display
  + The user (owner) who created the post
  + The comments related to the post
  + When a user's name is clicked, navigate to their profile page.

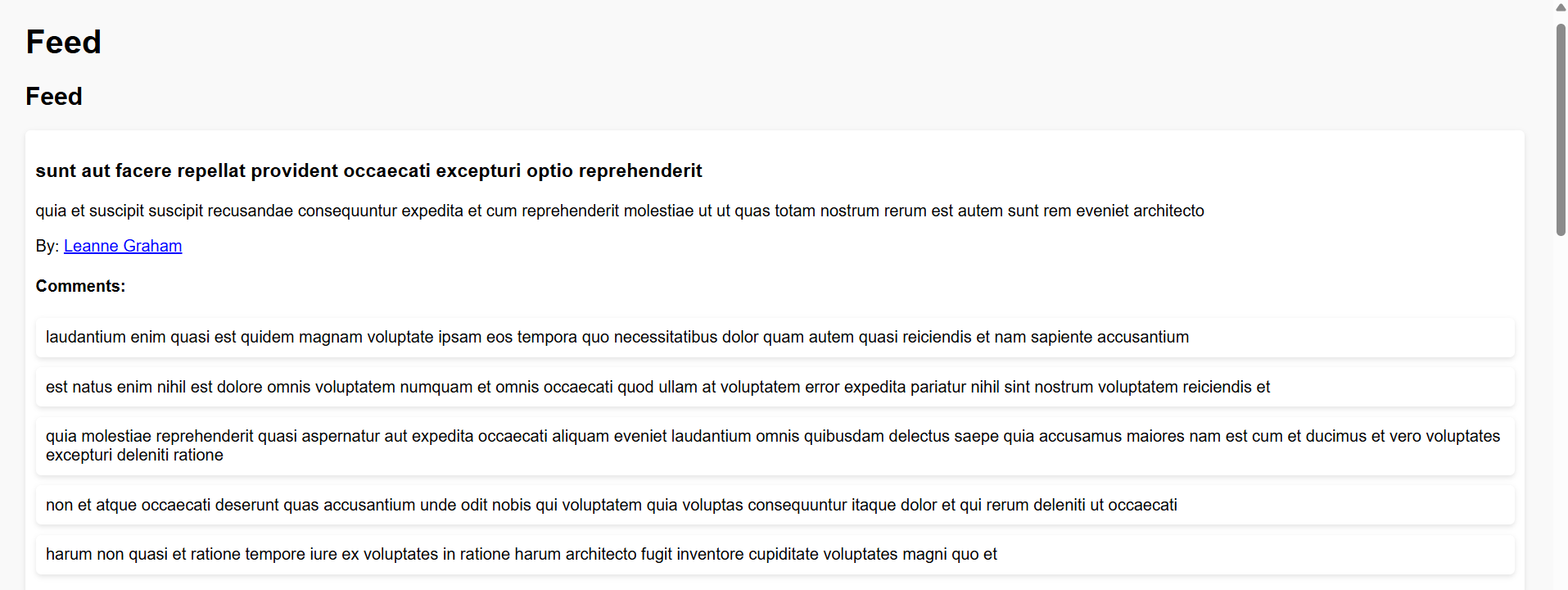
**User Profile View:**

* Display the basic details of the user
* Show a photo gallery of the user(**/albums** and **/photos API**)
* Show a list of todos associated with the user**(/todos api)**

**Hint**: *All entities (posts, comments, photos, todos) are interconnected using the userId(users). Use userId to associate and map the relevant data.*

Ans :

Screenshot of the page:



A screenshot of a computer

AI-generated content may be incorrect.

**Let there be two files in the same folder: *index.html* and *index.js*. Update code in index.js so that the counter app works perfectly.**

***index.html***

*<!DOCTYPE html>*

*<html lang="en">*

*<head>*

*<meta charset="UTF-8" />*

*<meta name="viewport" content="width=device-width, initial-scale=1.0" />*

*<title>Counter</title>*

*<script src="https://cdn.tailwindcss.com"></script>*

*</head>*

*<body class="bg-gray-100 h-screen flex items-center justify-center font-sans">*

*<div class="bg-white rounded-xl shadow-lg text-center py-10 px-16">*

*<div id="counter-value" class="text-6xl font-bold text-gray-800 mb-8">*

*0*

*</div>*

*<div class="flex justify-center gap-5">*

*<button*

*id="decrement"*

*class="py-4 px-8 text-lg bg-indigo-500 text-white rounded-lg transition duration-200 hover:bg-indigo-600 active:scale-95"*

*>*

*Decrement*

*</button>*

*<button*

*id="increment"*

*class="py-4 px-8 text-lg bg-indigo-500 text-white rounded-lg transition duration-200 hover:bg-indigo-600 active:scale-95"*

*>*

*Increment*

*</button>*

*</div>*

*</div>*

*<script src="./index.js"></script>*

*</body>*

*</html>*

***index.js***

*class Counter {*

*constructor() {*

*this.value = 0;*

*this.decrementBtn = document.getElementById("decrement");*

*this.incrementBtn = document.getElementById("increment");*

*this.counter = document.getElementById("counter-value");*

*this.counter.innerText = this.value;*

*}*

*increment() {*

*this.value++;*

*this.counter.innerText = this.value;*

*}*

*decrement() {*

*this.value--;*

*this.counter.innerText = this.value;*

*}*

*}*

*const counter = new Counter();*

Ans :

Screenshot:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.