**Câu 1:** Đáp án là D vì

Trong hàm chúng ta đã khai báo biến name với var. Điều đó có nghĩa là biến này sẽ được hoisted (một vùng nhớ sẽ được setup khi biến được khởi tạo) với giá trị mặc định là undefined, cho tới khi chúng ta thực sự định nghĩa biến đó. Trong hàm này, chúng ta chưa hề định nghĩa biến name tại dòng mà ta log ra, vậy nên giá trị mặc định của nó vẫn là undefined.

Các biến được khai báo với keyword let (và const) cũng được hoisted nhưng không giống như var, chúng không được khởi tạo. Chúng ta sẽ không thể truy cập chúng cho tới khi chúng ta khai báo (khởi tạo) chúng. Người ta gọi đó là "temporal dead zone". Khi ta truy cập đến một giá trị trước khi chúng được khai báo, JavaScript sẽ throws một ReferenceError.

**(từ khóa hoisting js)**

**Câu 2** Đáp án là A vì

Chúng ta có thể thấy là obj và obj2 đều đang tham chiếu đến cùng 1 obj, do vậy khi ta thay đổi giá trị obj thì kéo theo obj2 cũng thay đổi theo do vậy nó sẽ in ra kết quả là Hello.

**(từ khóa tham chiếu, tham trị obj)**

**Câu 3** Đáp án là D vì

* Đầu tiên từ khóa var và let không ảnh hưởng đến kết quả của biến trong hợp này
* Khi chúng ta sử dụng từ khóa new Number thì nó sẽ tạo ra 1 object Number, với các thông số liên quan => do vậy giá trị của b sẽ là object và object có giá trị là 3
* Khi so sánh == thì chúng ta chỉ so sánh giá trị do vậy dẫn đến kết quả của console.log() đầu tiên sẽ đúng
* Nhưng khi so sánh === chúng ta so sánh cả kiểu giá trị do vậy khi chúng ta thu được 2 console.log() cuối sẽ là kết quả false.

**Câu 4** Đáp án là A

Khi chúng ta không khai báo 1 từ khóa var, let hoặc const trước 1 biến, trong js sẽ ngầm định khai báo biến đó từ khóa var do vậy chúng ta thu được kết là {}

**(từ khóa tìm hiểu cách khai báo biến trong js)**

**Câu 5** Đáp án là B

Khi chúng ta chạy function demo với tham số truyền vào là number được khai báo ở ngoài = 10. Khi chúng ta truyền giá trị biến number vào funciton demo như vậy có nghĩa là biến number lúc này có phạm vi hoạt động chỉ trong function demo do vậy chúng ta có thay đổi biến number thế nào đi chăng nữa thì kết quả của biến number chỉ hoạt động trong funciton demo. Khi ra ngoài function demo thì kết quả biến number lại quay trở lại thành giá trị giống như đã khai báo trước function demo và = 10;

**Câu 6:** Đáp án là D vì

Khi chúng ta cộng 1 số với 1 chuỗi thì js sẽ convert số của chúng ta thành dạng String do vậy phép cộng bây giờ trở thành là phép cộng chuỗi

**Câu 7** Đáp án là A

Với từ khóa var bạn có thể khai báo bao nhiêu biến trùng tên tùy thích. Biến đó sẽ có giá trị là lần cuối khai báo.

Bạn không thể làm điều tương tự với let hay const vì chúng là *block-scoped*.

**Câu 8** Đáp án là C

Nếu chúng ta có 2 keys giống nhau, thì chúng sẽ bị replace. Nó sẽ vẫn nằm ở vị trí đầu tiên chúng được khai báo, nhưng giá trị thì sẽ là giá trị lần cuối

**Câu 9** Đáp án là E

Vì đây là phép toán hoặc do vậy chỉ cần 1 trong các điều kiện trả về giá trị đúng thì nó sẽ lấy luôn giá trị đúng và không lấy các giá trị đằng sau nữa, trong trường hợp tất cả các giá trị đều sai thì biến sẽ lấy giá trị cuối cùng

**Câu** 10 Đáp án là C

Vì chúng ta đang gán giá trị của mảng vào 1 mảng mới theo lần lượt các vị trí. Do chúng ta [y] do vậy chúng ta chỉ lấy được giá trị đầu của mảng