**FUNCTION, SCOPE, CLOSURE**

**Function – Hàm**

Là khối mã có thể tái sử dụng.

3 cách khai báo:

A computer screen shot of a computer code

AI-generated content may be incorrect.

**Scope – Phạm vi biến**

| **Loại Scope** | **Mô tả** |
| --- | --- |
| **Global** | Biến được truy cập ở mọi nơi |
| **Function** | Biến chỉ dùng được trong hàm |
| **Block** | let, const chỉ dùng được trong {} |

A screen shot of a computer program

AI-generated content may be incorrect.

**Closure – Hàm ghi nhớ phạm vi**

Closure = function + scope xung quanh khi nó được tạo ra.

A computer screen shot of a program code

AI-generated content may be incorrect.

inner() vẫn nhớ biến counter dù outer() đã chạy xong → Đây chính là **closure**

**Câu hỏi luyện tập (trắc nghiệm nhanh)**

1. Hàm nào sau đây là arrow function?  
   a. function hello() {}  
   b. const hi = () => {} ✅  
   c. var greet = function() {}  
   d. function() => {}
2. let x = 10; khai báo trong {} thuộc phạm vi nào?  
   → **Block scope**
3. Closure dùng để làm gì?  
   → Ghi nhớ các biến tại thời điểm function được tạo ra.

**Mini Quiz – Kiểm tra hiểu bài**

**Câu 1:** Đoán kết quả:

A computer screen with text on it

AI-generated content may be incorrect.

**Câu 2:**

A screenshot of a computer

AI-generated content may be incorrect.

**Câu 1:**

* 1 và 2 → vì count vẫn được giữ nhờ closure.

**Câu 2:**

* 2 và 1 → a trong foo là biến riêng.

</details>

**Phỏng vấn thường hỏi**

**Q:** Closure là gì?  
**A:** Closure là khi một hàm nhớ được các biến trong phạm vi mà nó được tạo ra, dù cho hàm cha đã kết thúc.

**Q:** Khác biệt giữa function declaration và arrow function là gì?  
**A:** Arrow function không có this, không dùng được làm constructor, cú pháp ngắn hơn.

**Bài tập nhỏ #2: Viết Closure đơn giản**

Tự viết function createCounter() mà mỗi lần gọi next() sẽ tăng số đếm lên:

A black screen with red and white text

AI-generated content may be incorrect.