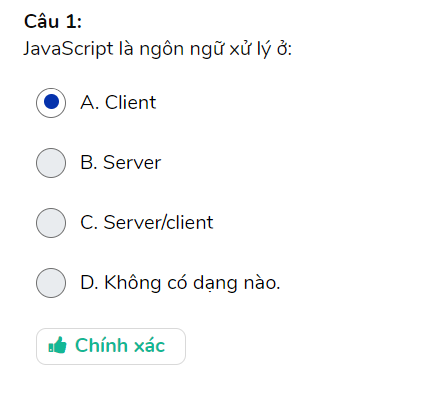
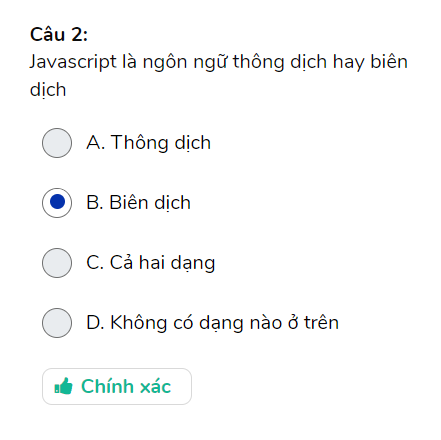
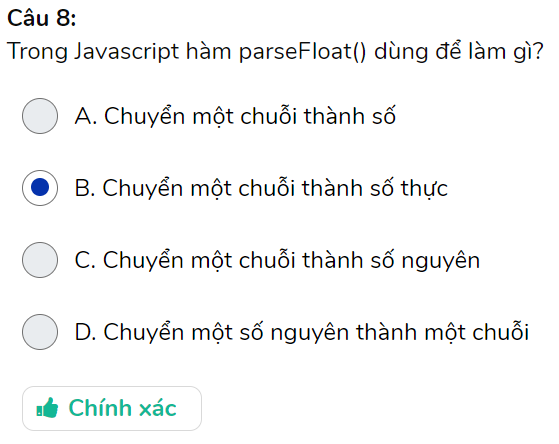
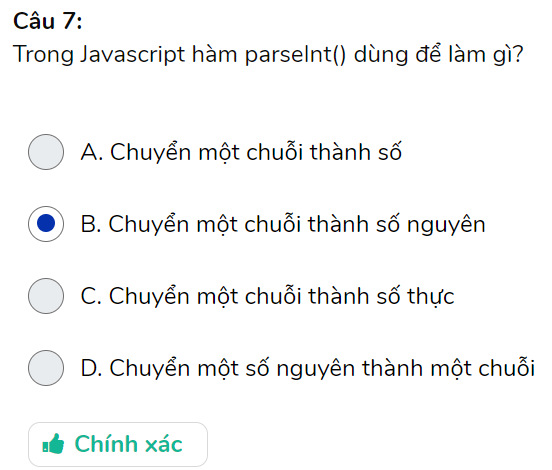
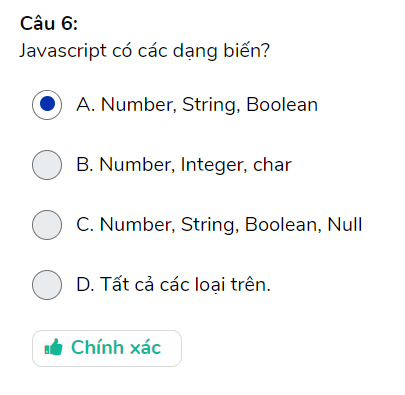
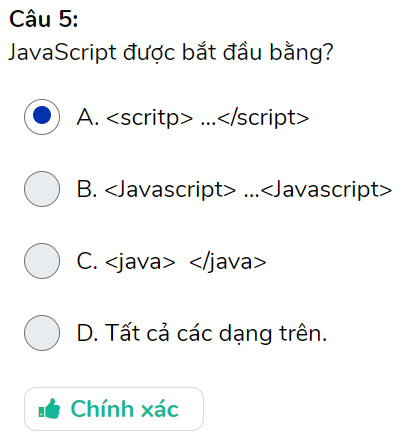
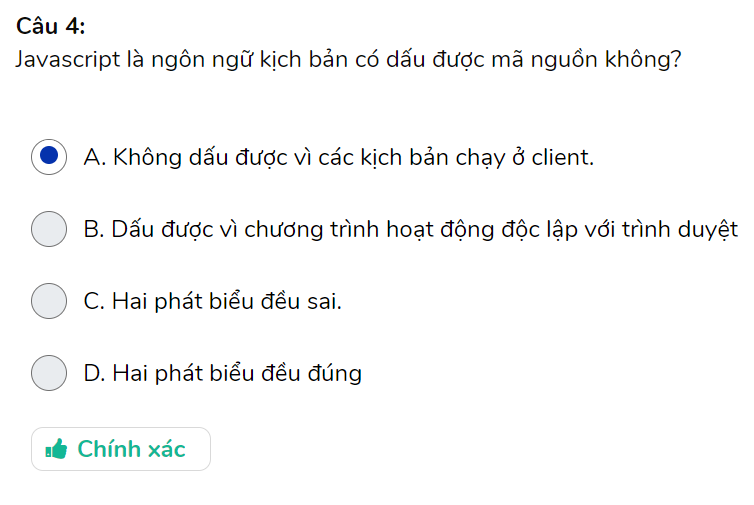
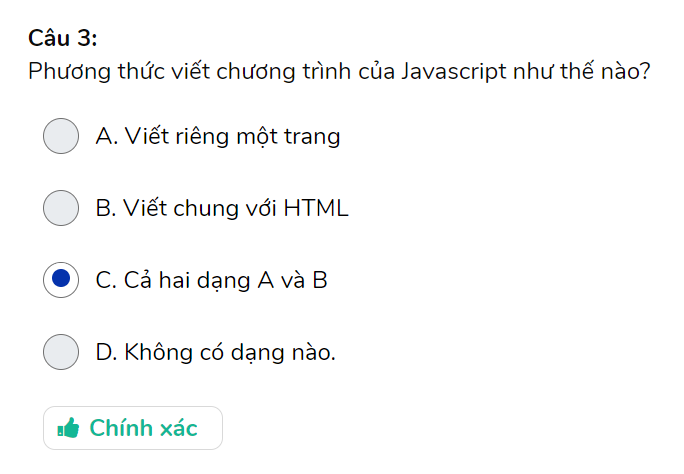
ĐỀ WEB

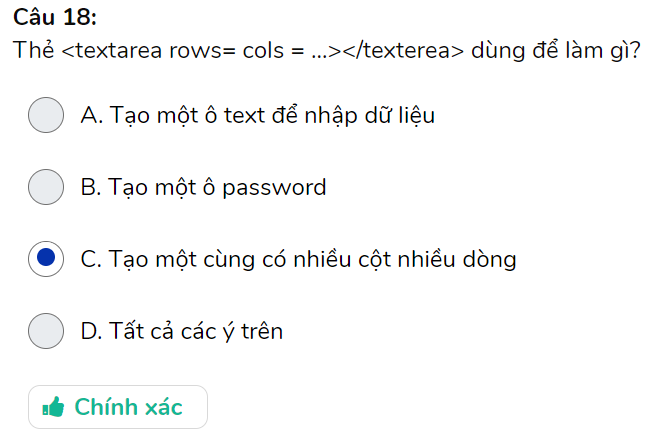
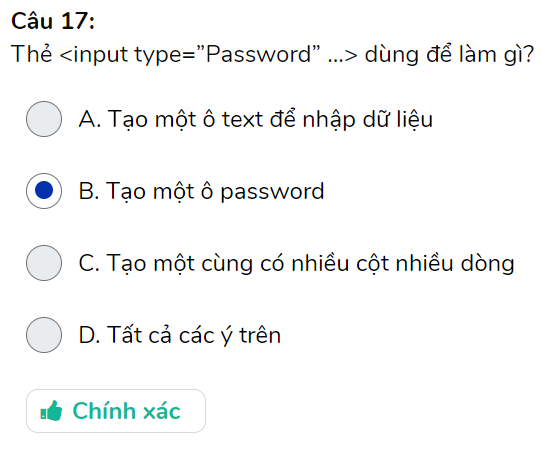
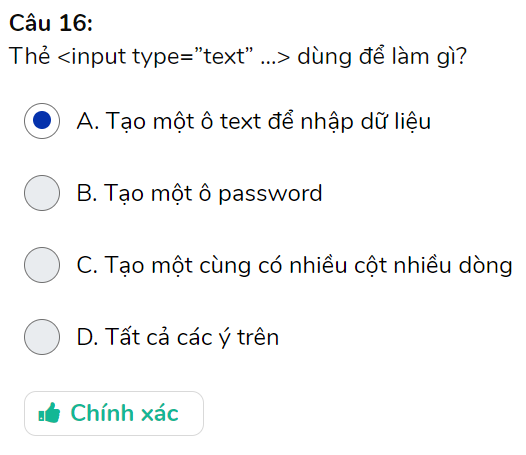
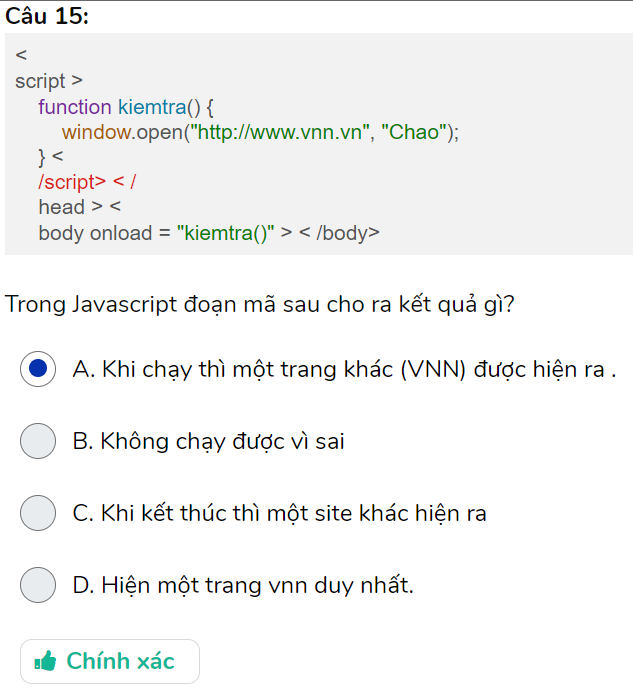
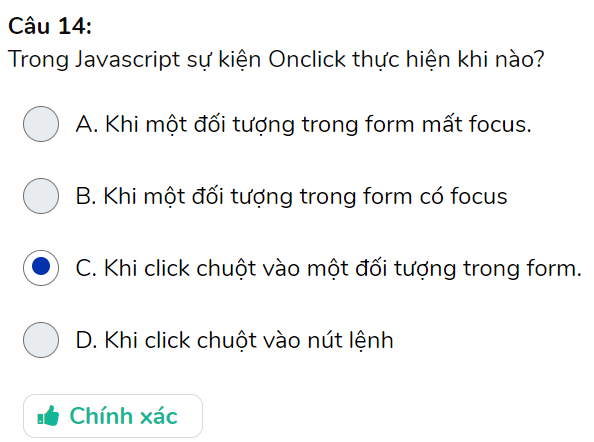
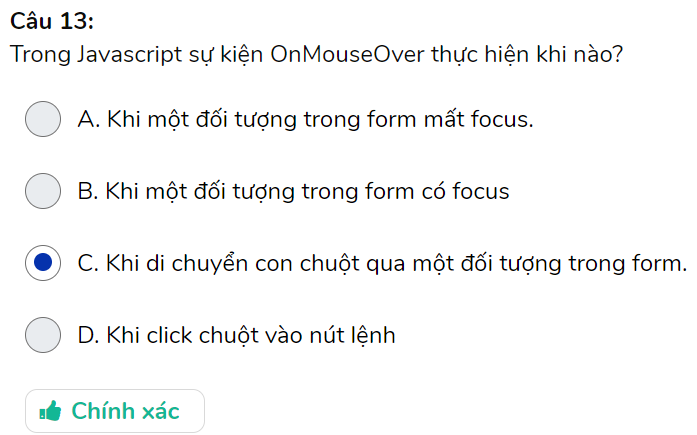
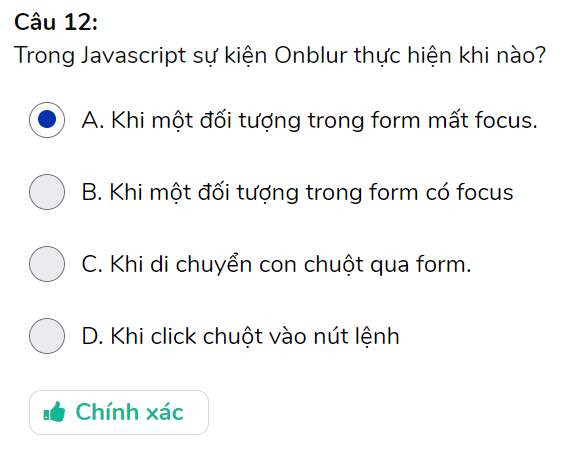
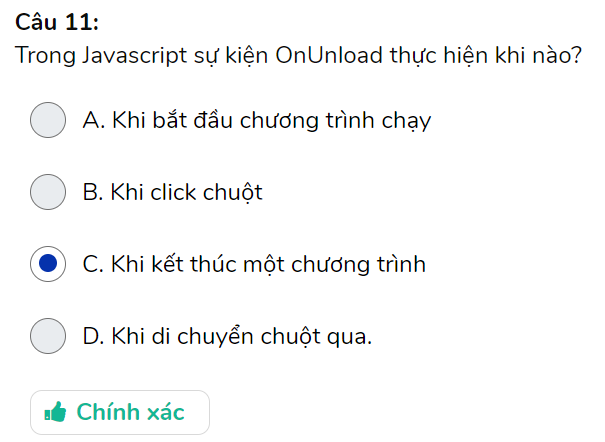
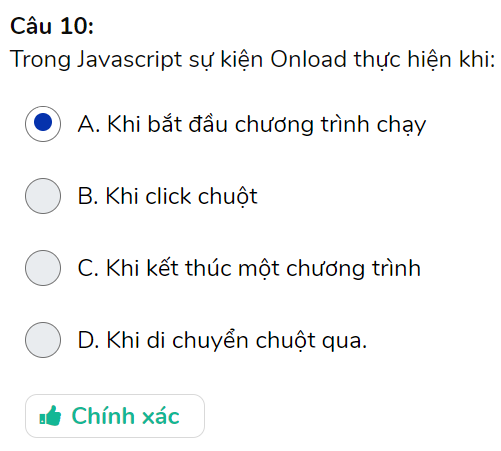
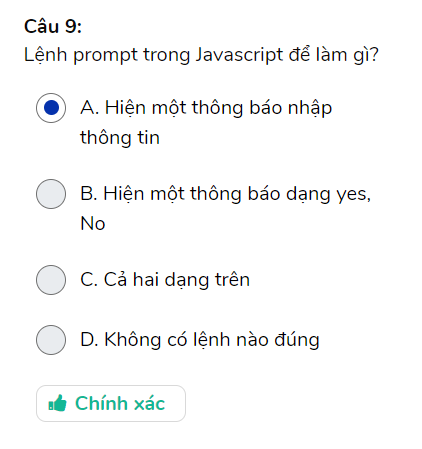
JAV

Phần 1:

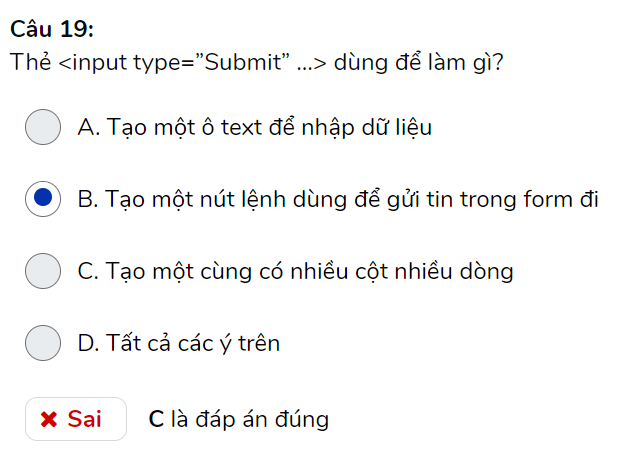


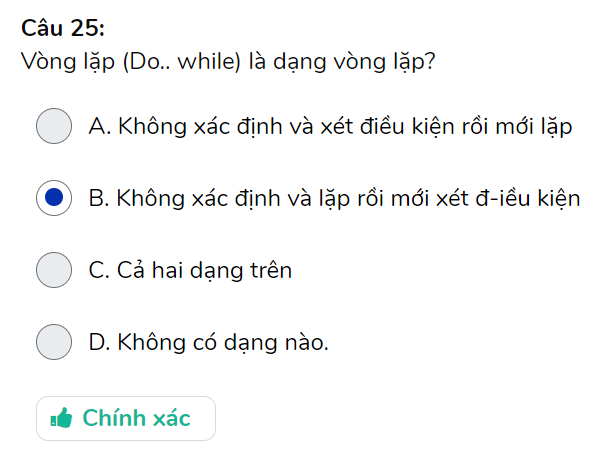
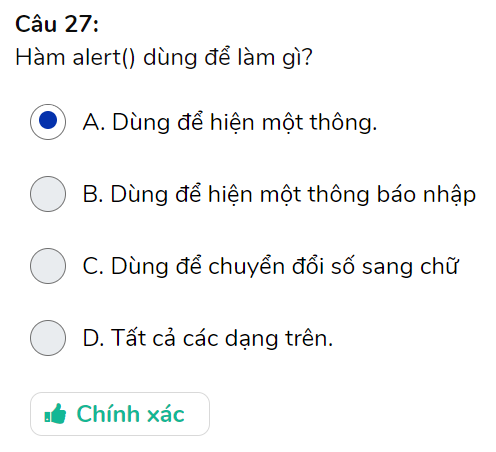
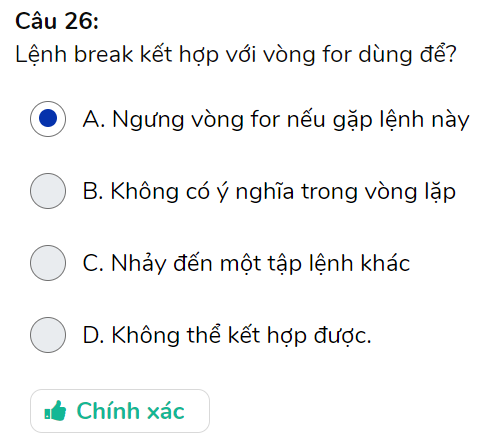
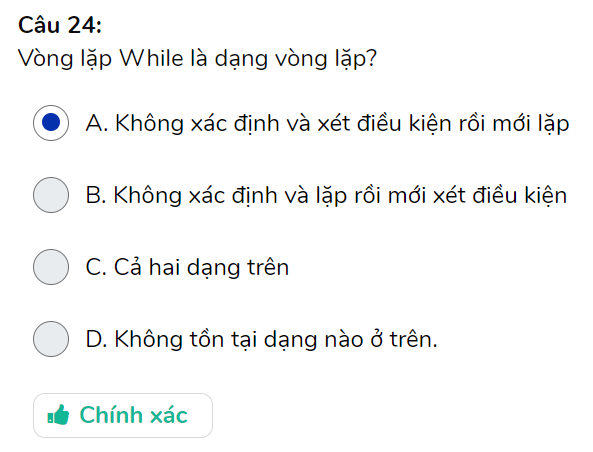
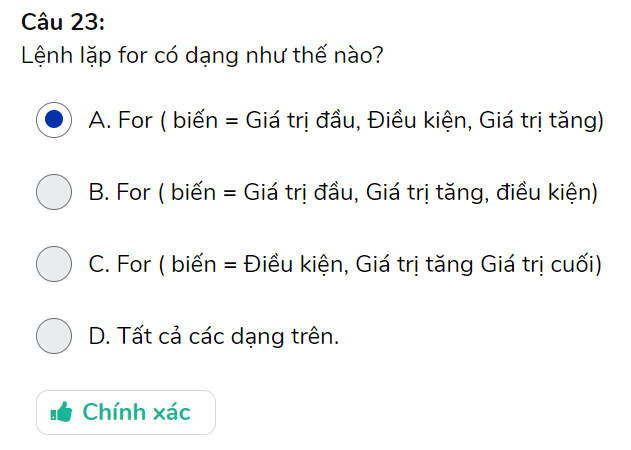
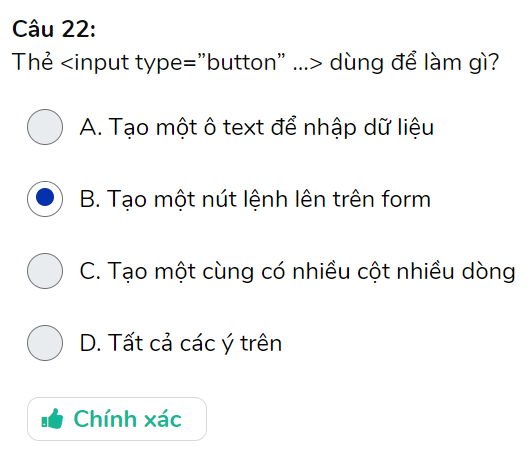
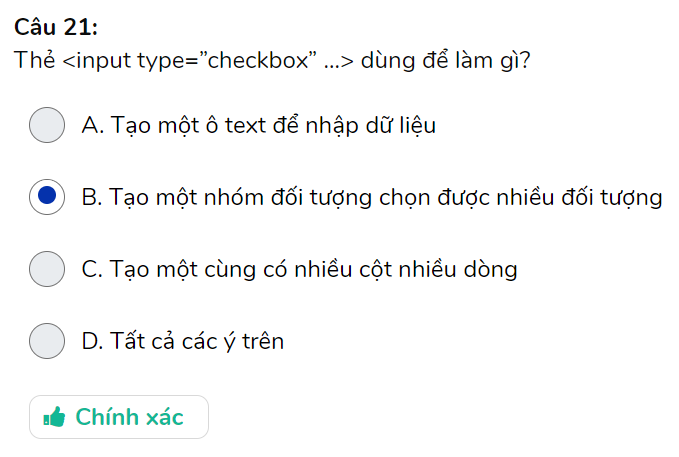
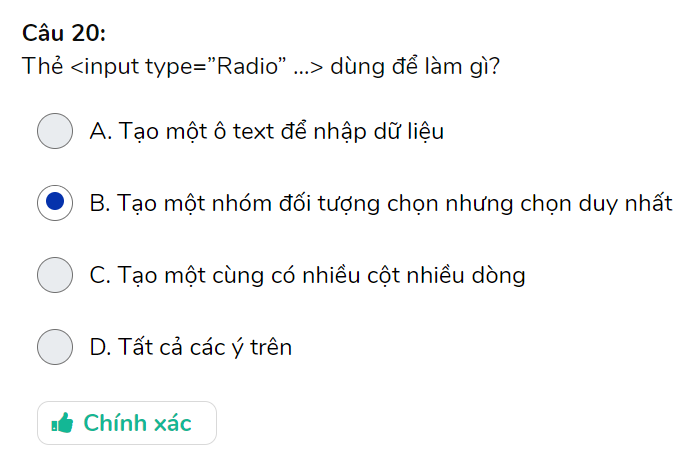


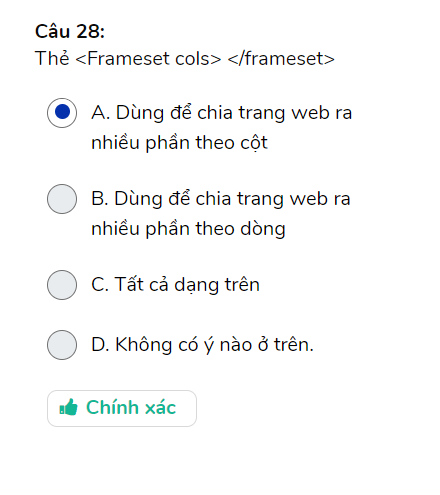


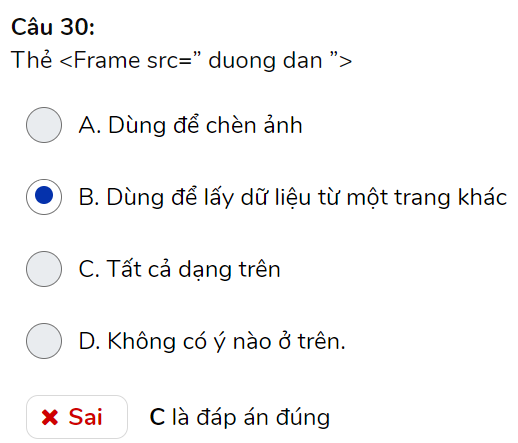
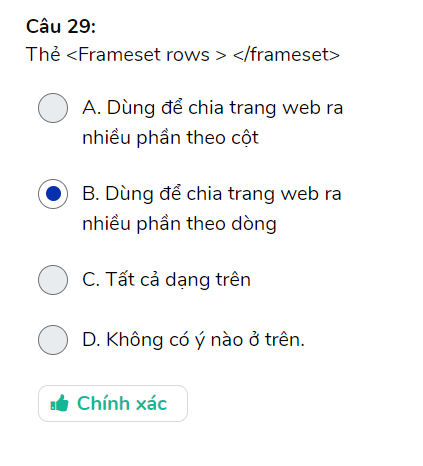


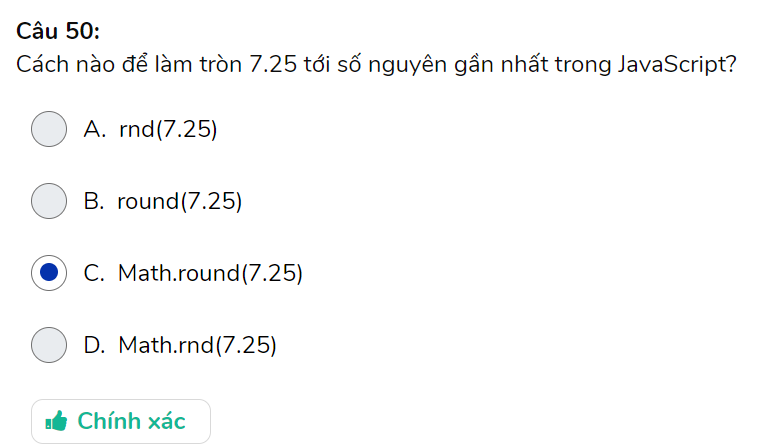
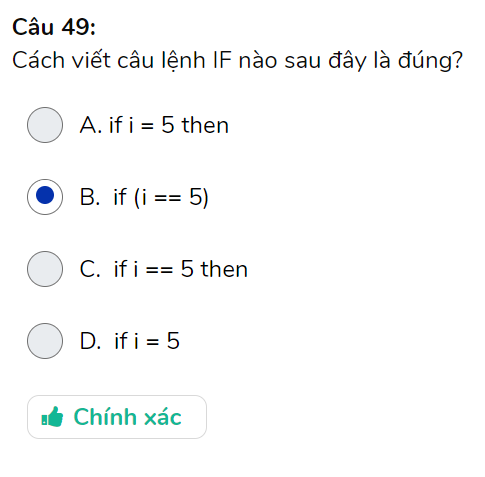
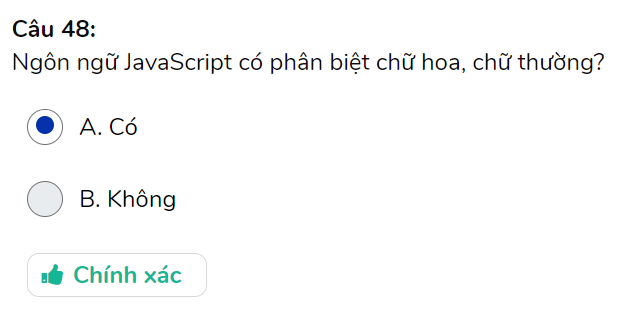
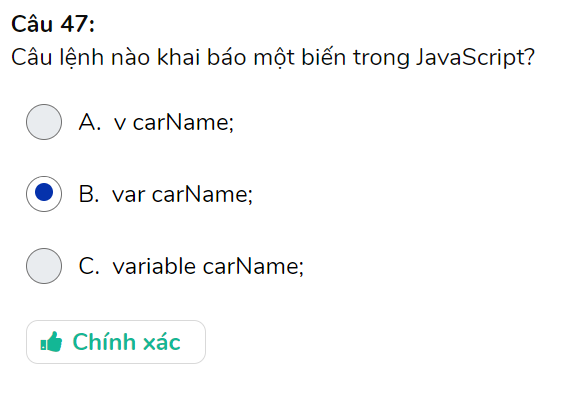
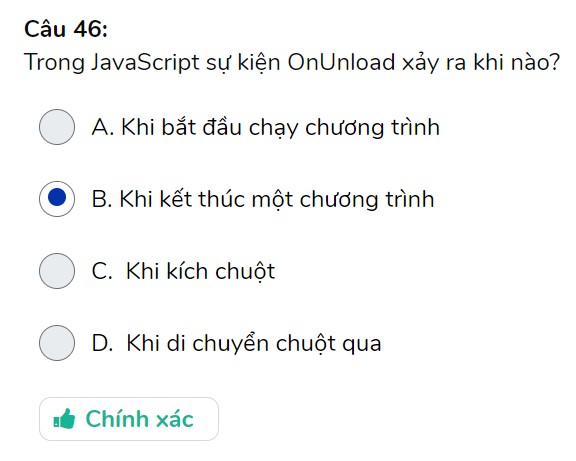
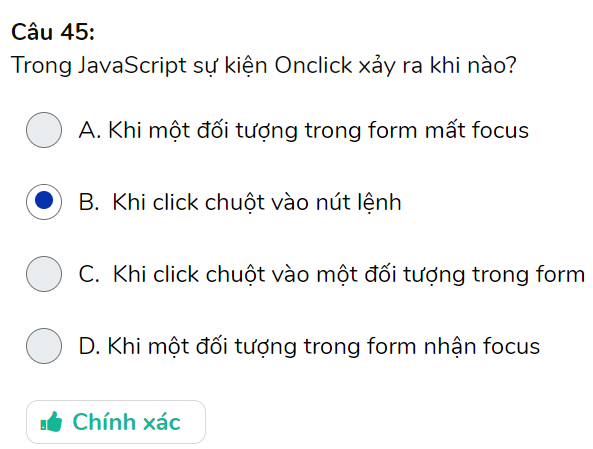
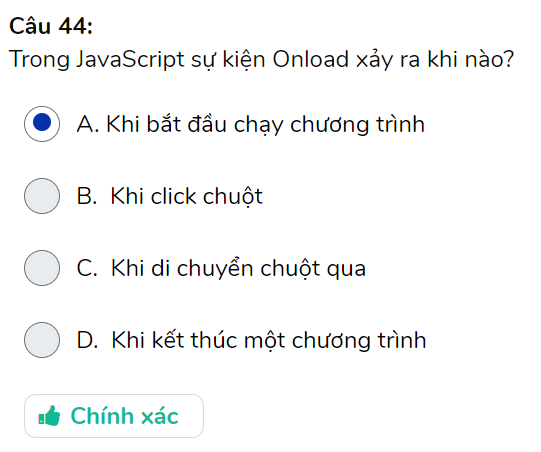
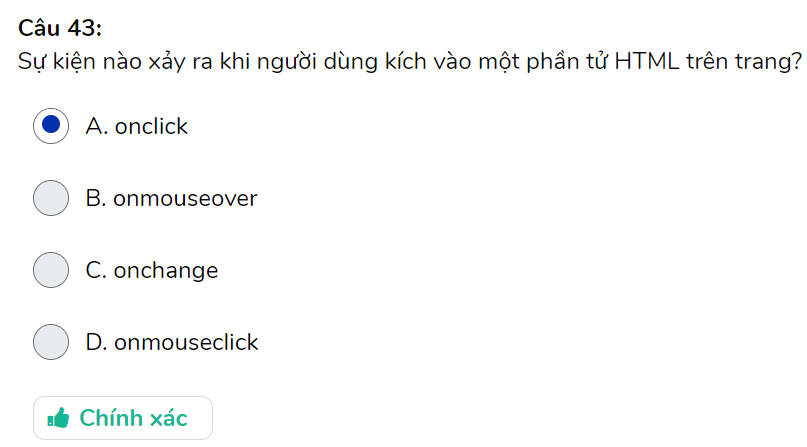
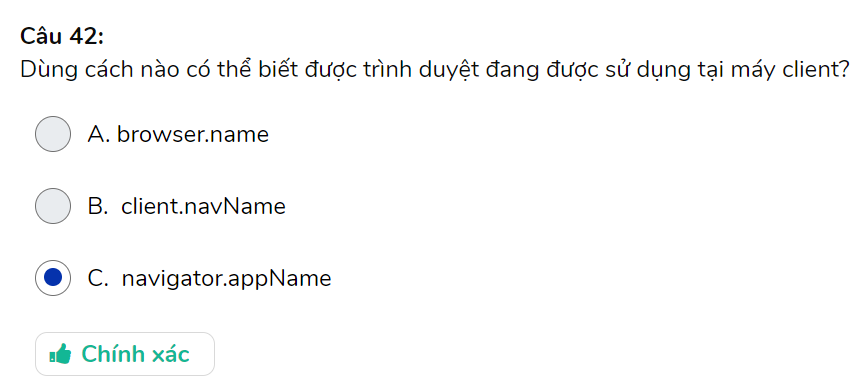
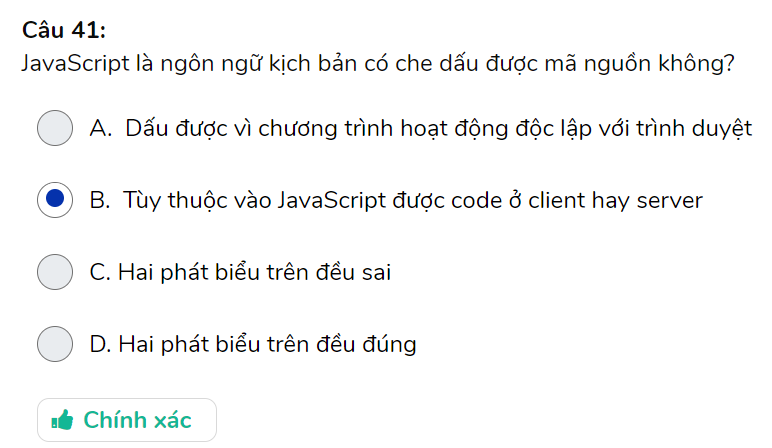
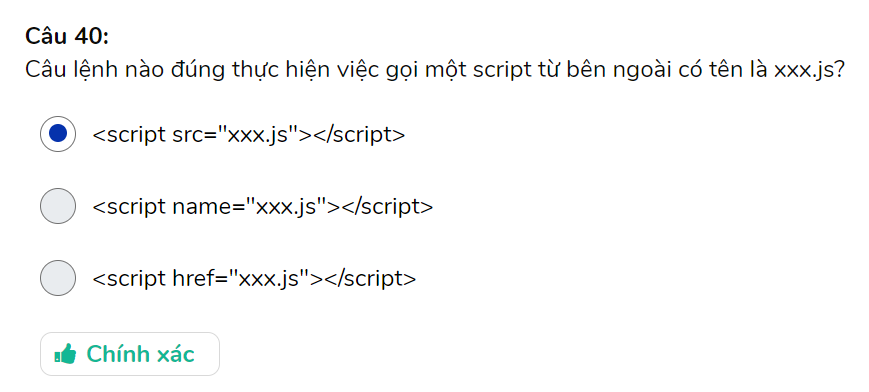
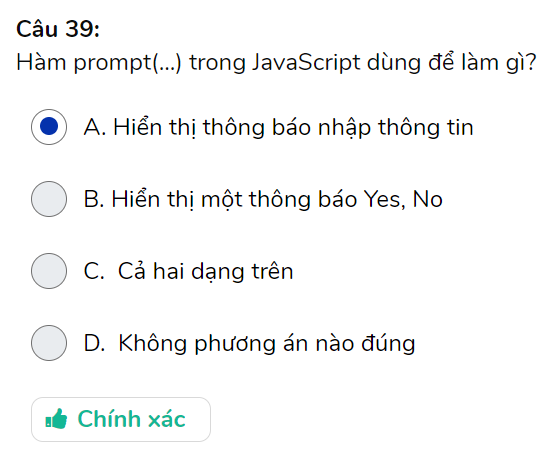
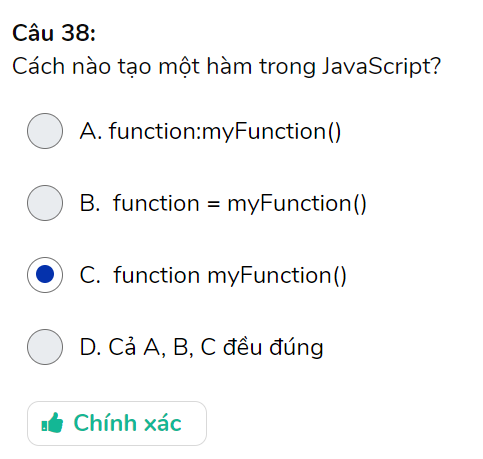
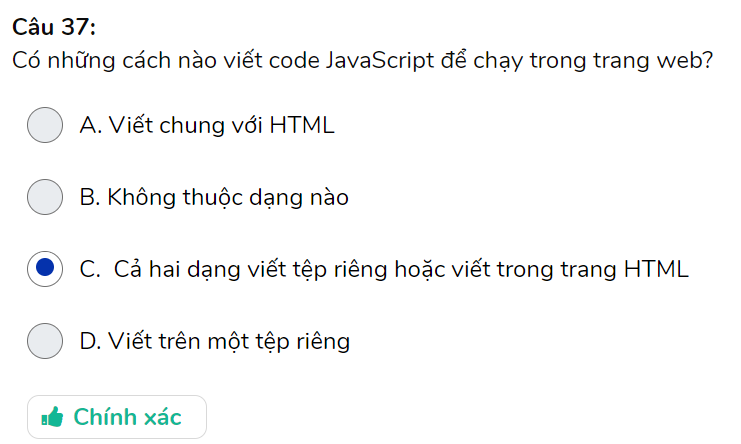
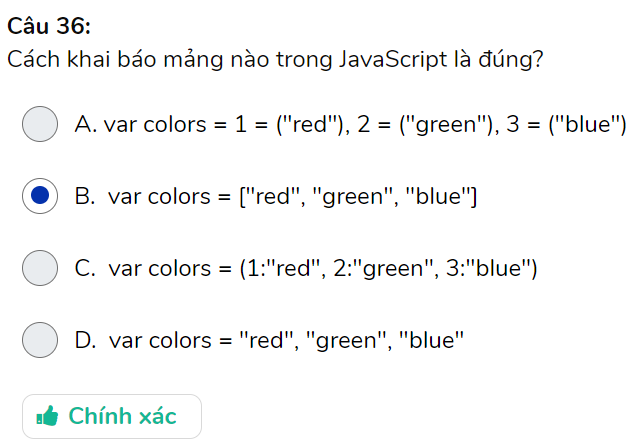
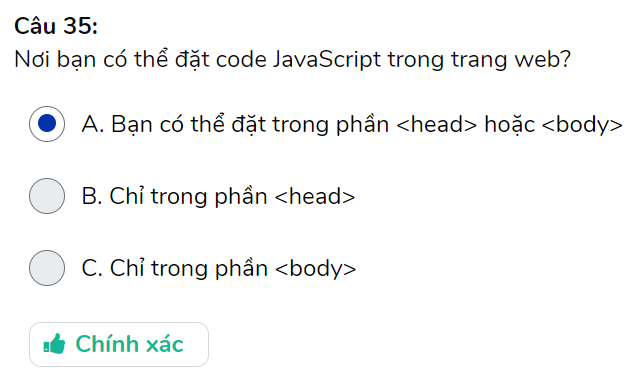
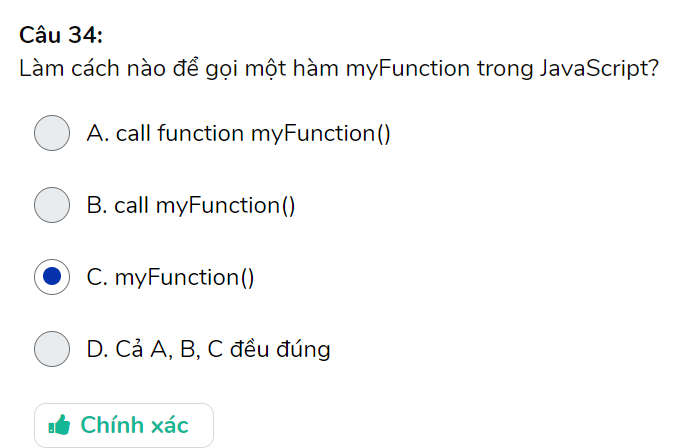
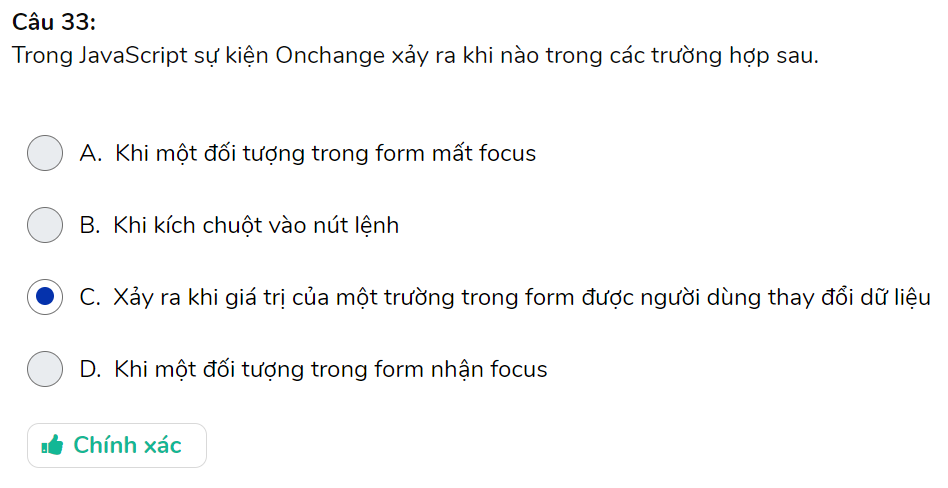
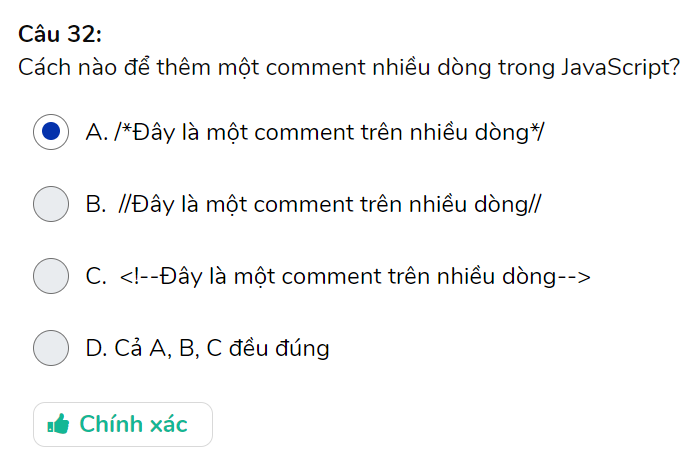
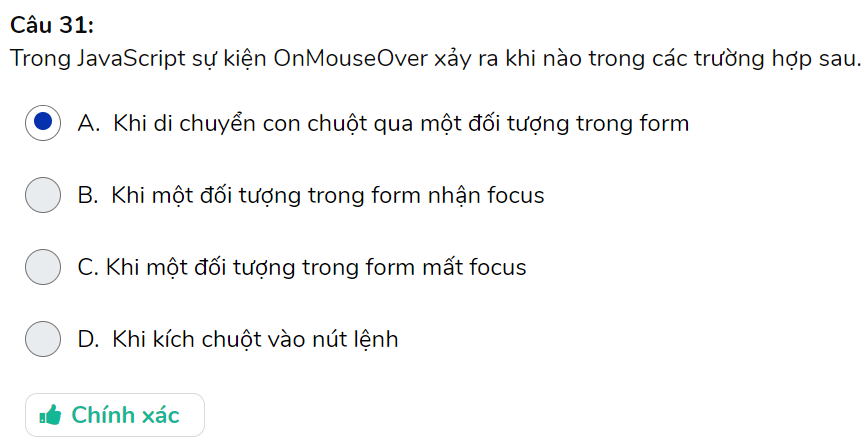
\*vùng



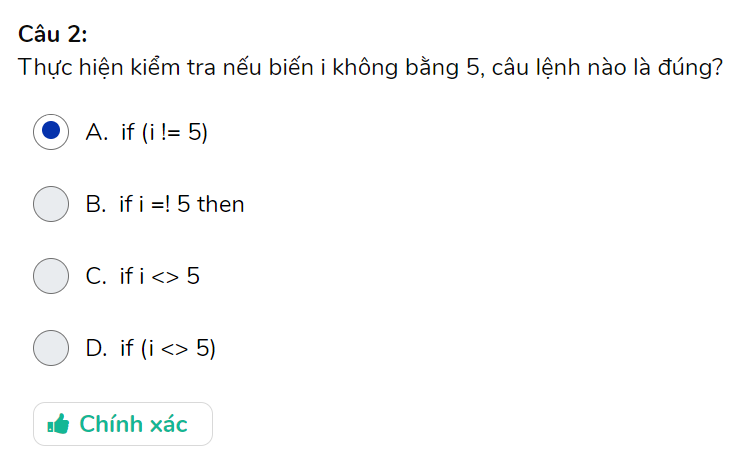
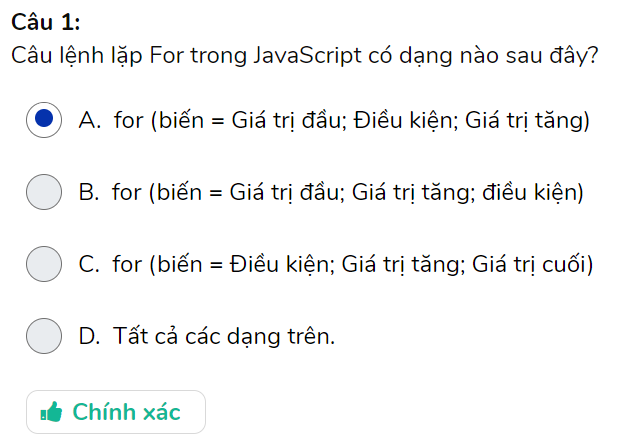
\*mé bịp

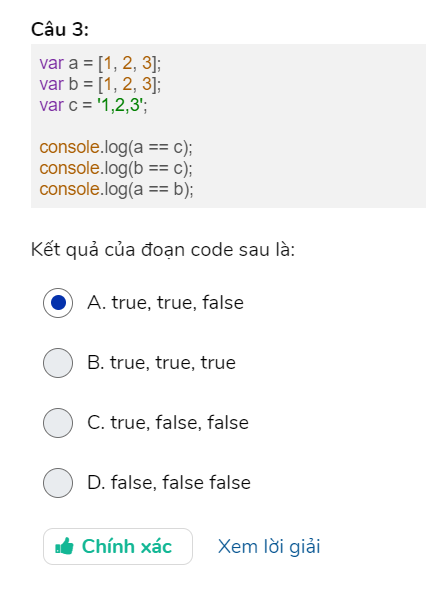




\*bịp (30.B nha), /// chèn ảnh được mà ba : D 

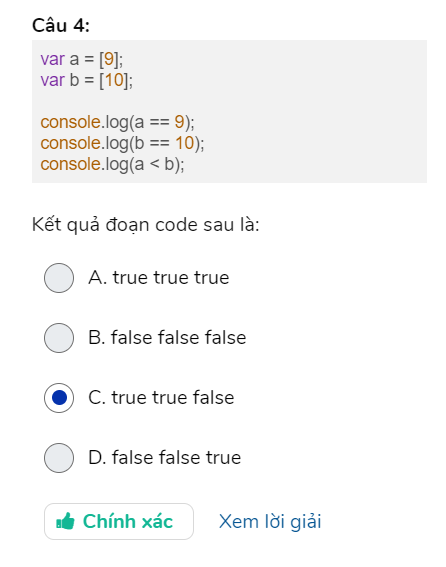
Phần 2:





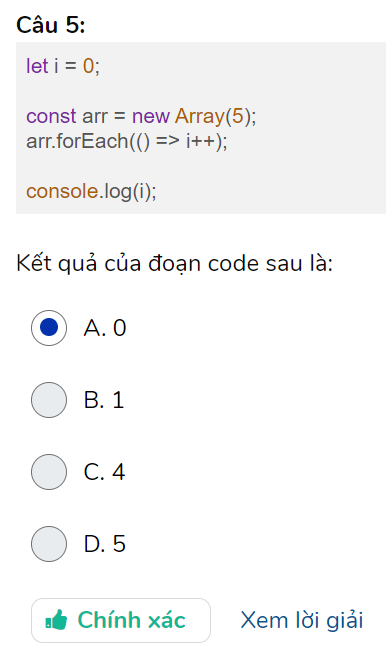
Khi so sánh Double Equals (==) giữa array và string, cụ thể là a hoặc b với c, Javascript sẽ tự động gọi arr.toString() để chuyển đổi array sang string trước khi so sánh, hai mảng a và b convert sang string sẽ là '1,2,3', vì thế a == c và b == c cho kết quả true.

Khi so sánh Double Equals (==) hay Triple Equals (===) giữa các đối tượng là kiểu dữ liệu tham chiếu (Reference Type), như object, array, function, chúng ta không quan tâm đến giá trị mà đối tượng đang chứa, mà chỉ quan tâm đến chúng có cùng trỏ đến một địa chỉ ô nhớ hay không mà thôi. Trong trường hợp này, a và b là hai array trỏ đến hai ô nhớ khác nhau, vì thế a == b cho kết quả false.

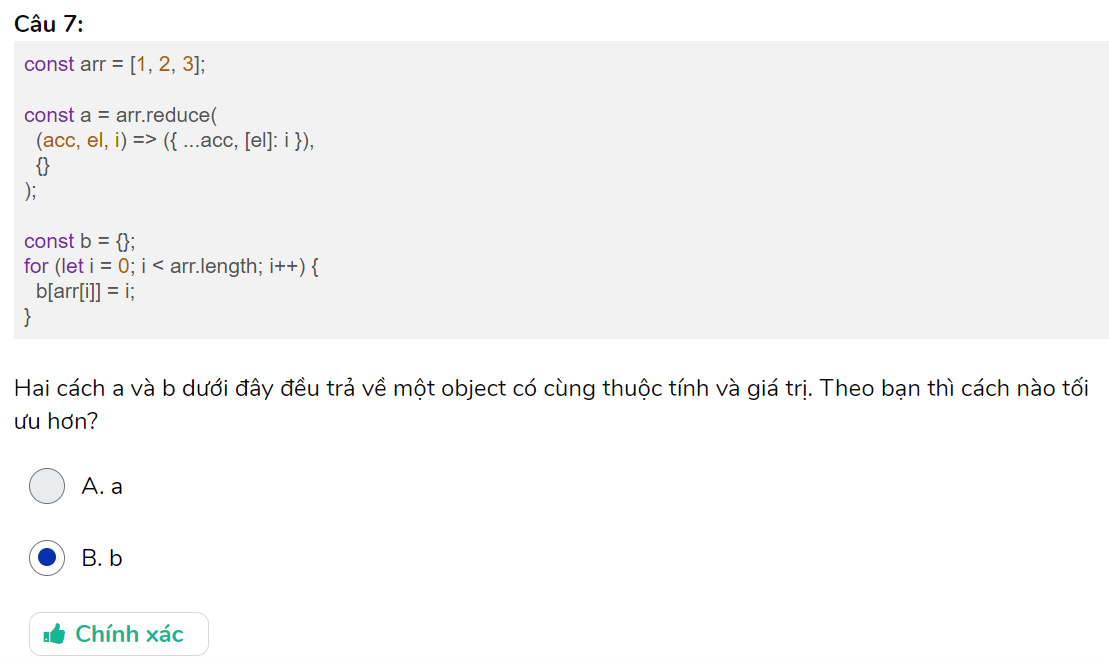


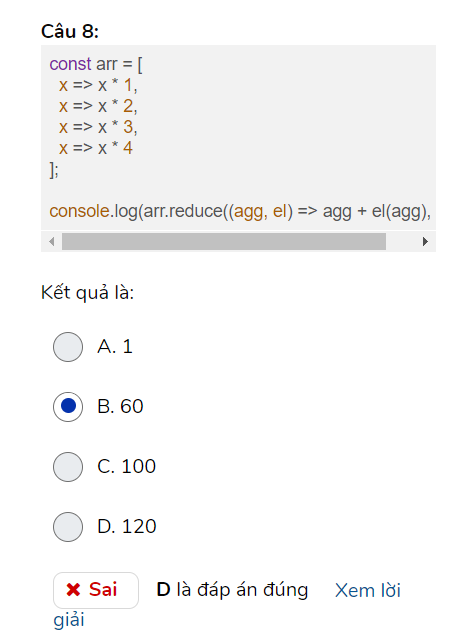
Khi so sánh Double Equals (==) giữa array và number, Javascript sẽ chuyển đổi array sang number trước khi so sánh ([9] -> 9 và [10] -> 10), vì thế [9] == 9 và [10] == 10 cho kết quả true.

Nhưng khi so sánh hai array với toán tử < hoặc >, lúc này array sẽ không được chuyển đổi sang number mà là sang string ([9] -> "9" và [10] -> "10"). Khi so sánh hai string thì sẽ so sánh theo alphabet với từng ký tự một, vì thế "9" < "10" cho kết quả là false vì "9" < "1" là sai.









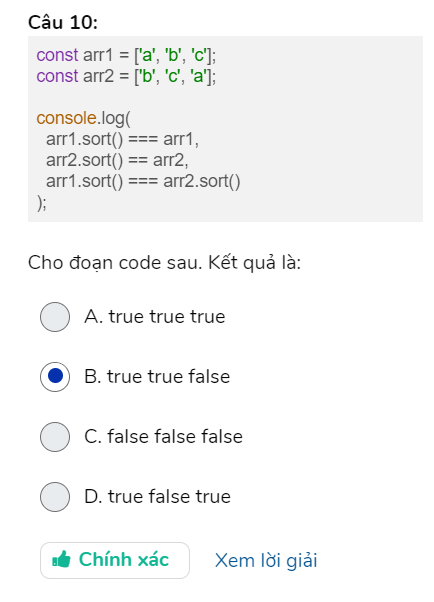
Hàm reduce của array cho phép chúng ta truyền vào một giá trị ban đầu ở tham số thứ hai, trong trường hợp này là 1 và ta có các bước tính toán sau:

1 + 1 \* 1 = 2

2 + 2 \* 2 = 6

6 + 6 \* 3 = 24

24 + 24 \* 4 = 120



Hàm sort sẽ sắp xếp lại array và đồng thời trả về chính tham chiếu đến array đó. Vì vậy arr1.sort() và arr1 tham chiếu đến cùng một object trong bộ nhớ, điều này cũng đúng cho arr2.sort() và arr2.

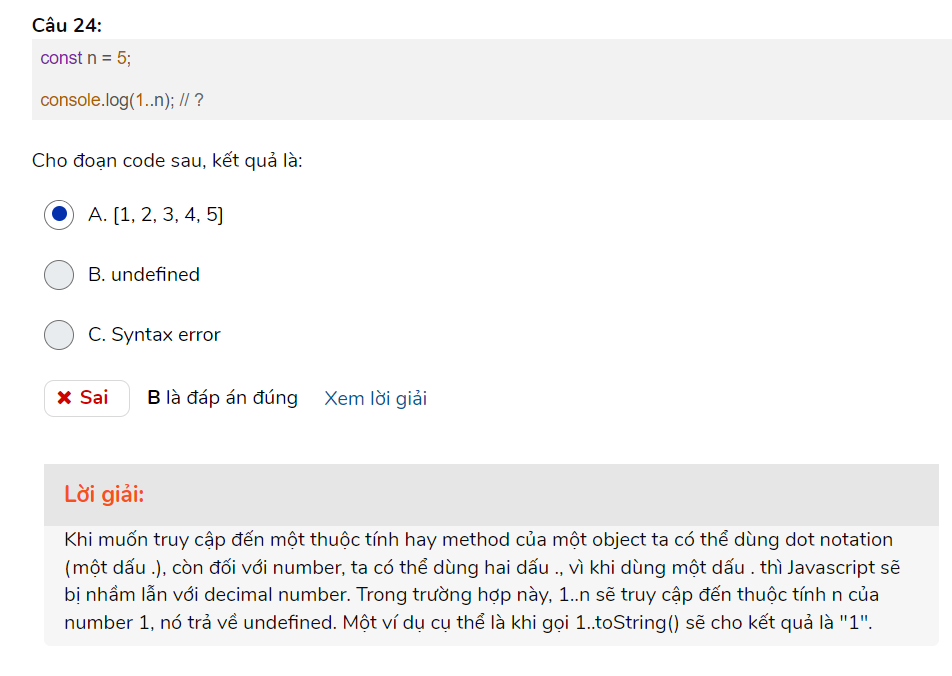
Với arr1.sort() và arr2.sort() thì rõ ràng chúng tham chiếu đến hai object khác nhau trong bộ nhớ.

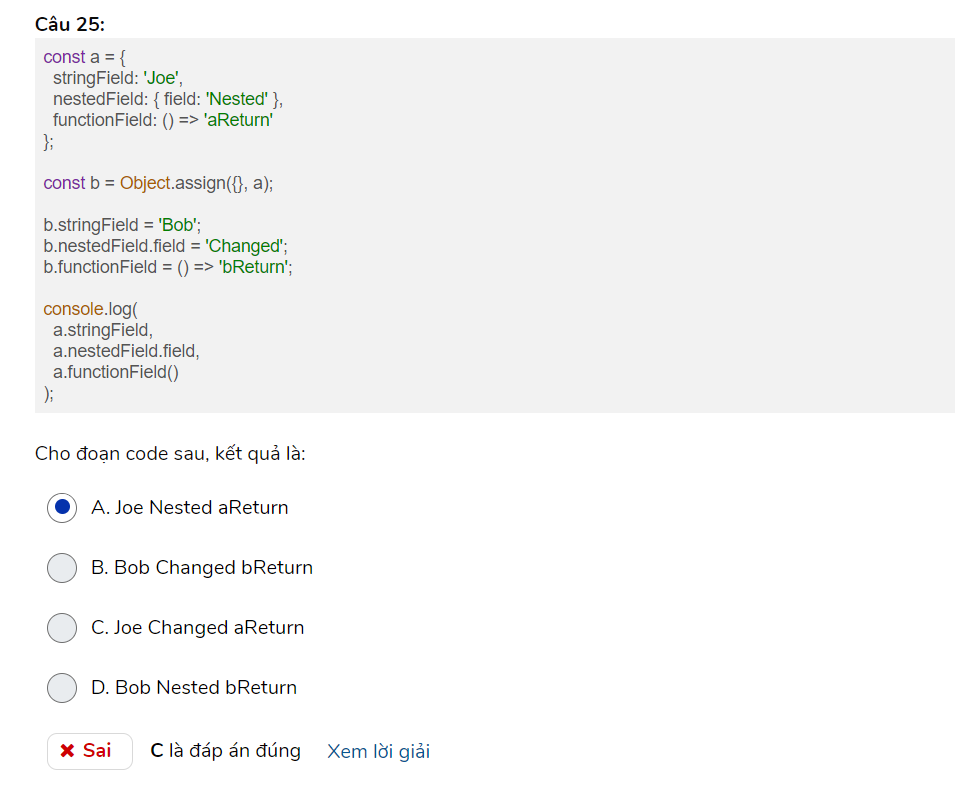


Khi gọi một Function Constructor, this sẽ được tạo và trả về một cách ngầm định nếu chúng ta gọi bằng từ khóa new, nếu không this sẽ không được tạo và sẽ là global window object (trong trình duyệt).

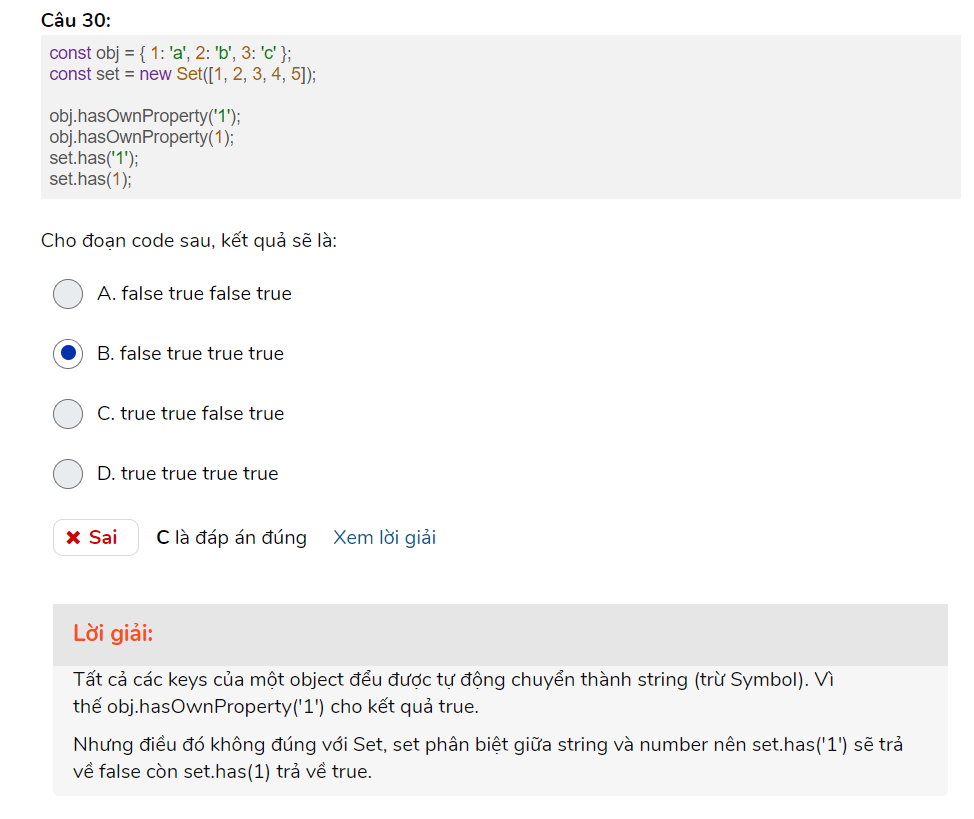


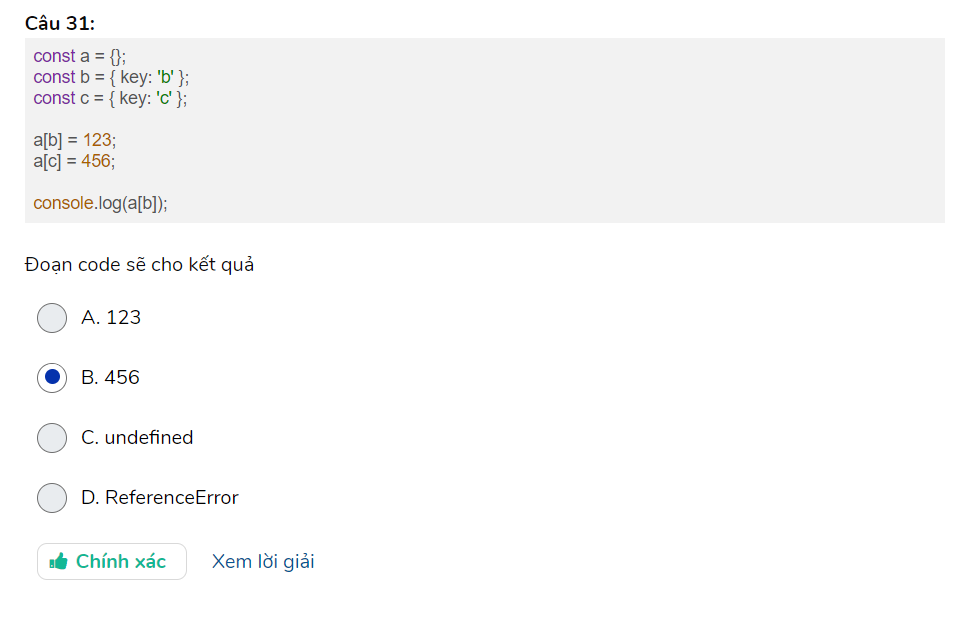
Trong Javascript, khi sử dụng Triple Equals (===) thì null và undefined chỉ cho kết quả true khi so sánh với chính nó, NaN thì luôn cho kết quả false khi so sánh với bất kỳ object nào, kể cả chính nó, còn [NaN] là một array bình thường chỉ chứa một phần tử là NaN.





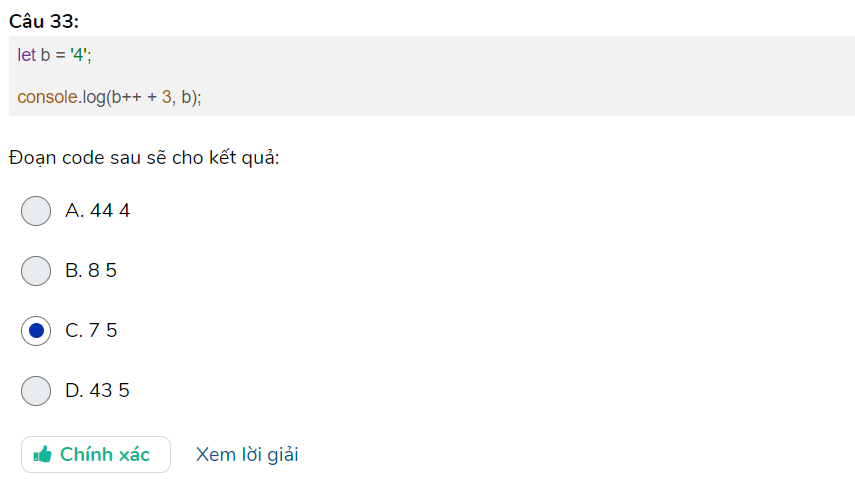
b = Object.assign({},a); sẽ thực hiện một shallow copy trên object a, bất kỳ thuộc tính nào của b là object đều tham chiếu đến cùng thuộc tính trong a. Vì vậy khi chúng ta thay đổi nested field của b, thì nested field của a cũng thay đổi theo.



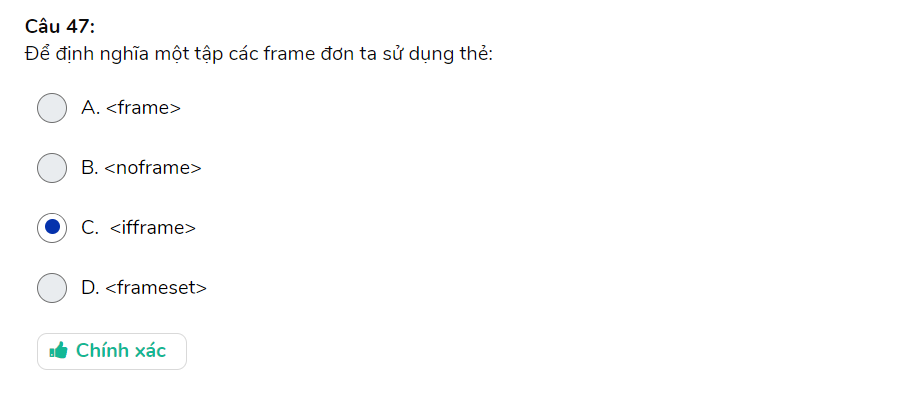


Tất cả các keys của một object đểu được tự động chuyển thành string (trừ Symbol).

Khi một object chuyển sang string, nó có giá trị "[object Object]", vậy a[b] = 123 có thể viết thành a["object Object"] = 123, tương tự với a[c] = 456 sẽ là a["object Object"] = 456.







------------------------------------

PHP