**Problem 1:**

Output: []

In “empty” function, the parameter “o” is reference to an array object; if we assign null to “o”, “o” reference to null object. So that, “x” still reference the original array.

**Problem 2:**

Output: 1

In the code, function “swap” have 2 argument x and y which have number valuem, so the two parameter “a” and “b” of “swap” function have a number value.

So that, in the “swap” function, it just swap the value of “a” and “b” and not effect of “x” and “y” variable.

**Problem 5:**

Output:

Undefine

2

Because the **hoisting** mechanism, the declaration of “a” and fucntion “foo” are moved to the top of function test.

At line 13th, “a” is declared but not be initialized, so it log “Undefine” to the screen.

At line 14th, function foo is declared, so it log “2” to the screen.

**Problem 6:**

Output: 5

In “someFunction”, “a” is hoisting to the top, so “otherFunction” can access to “a” no mater where it will be call because **the lexical scoping**.

At line 33th, “someFunction” return “otherFunction” for “firstResult”, so “firstResult” contain a function. After that, it call the function and store the result to “result” which value is 2.

**Problem 7:**

Output: 1

Because in b function, a is function and it hoisting to the top of b, so that a is redeclare and when a is assigned to 10, the “a” variable is not change.

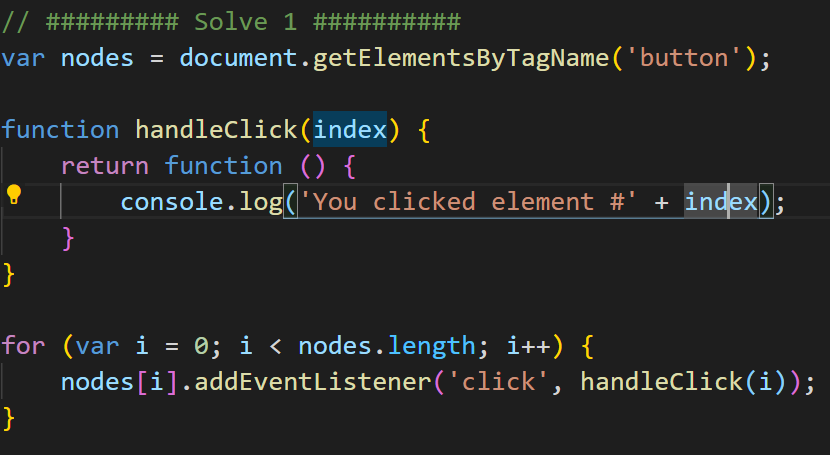
**Problem 8:**

It will log “You clicked element #4” for every button.

In the for loop, “i” is declare by var key word, so is hoiting to the top of main.js file.

Because the **closure** of the anonymous function at line 45th , is contain the “i” which is have value is 4 at the end of the for loop. When we click the button the current value of “i” is 4.

To solve this problem, we can use a variable contain value of “i”, and that variable is closuse to even handle function in other function.



We can also use let in for loop instead.

