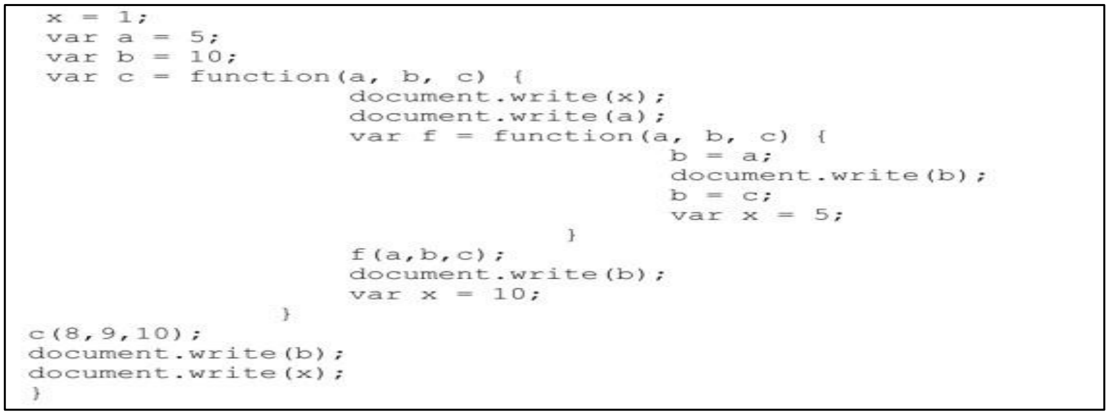
Lab 6

1. What will be printed by the following code snippets (answer without running it)?
2. Draw the lexical environment diagram for each code snippet.



```
 x = 1;
 var a = 5;
 var b = 10;
 var c = function(a, b, c) {
                  document.write(x);
                  document.write(a);
                  var f = function(a, b, c) {
                                    b = a;
                                    document.write(b);
                                    b = c;
                                    var x = 5;
                              }
                  f(a,b,c);
                  document.write(b);
                  var x = 10;
              }
c(8,9,10);
document.write(b);
document.write(x);
}
```

Code 1

LE: Lexical Environment

TDZ: Temporary Dead Zone

LE Global creation:

{outer:null, a : undefined, b : undefined, x: undefined, c : undefined}

LE Execution:

{outer : null , a :5 , b : 10, c : fn, x: 1, c : fn }

c LE creation:

{ outer : global, a: undefined, b: undefined, c : undefined, arguments [‘0’ : 8 , ‘1’ : 9 , ‘2’ : 10] , x : undefined }

c LE execution:

{ outer : global, a:8, b:9, c:10 , arguments [‘0’ : 8 , ‘1’ : 9 , ‘2’ : 10] , x: 10, f : fn }

f LE creation:

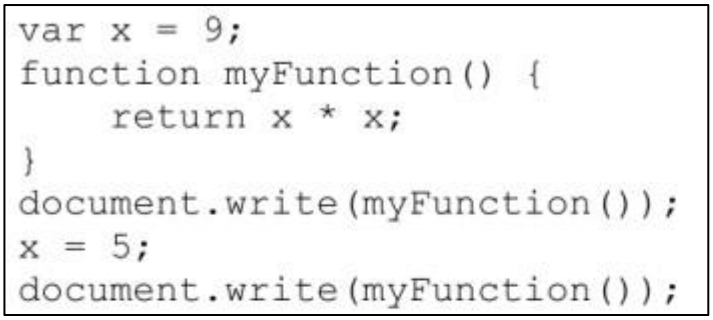
{outer : c , a: undefined, b : undefined, c : undefined, x : undefined}

f LE execution:

{outer : c , a: 8, b : 10, c : 10, x : 5}

Output:

```
undefined
8
8
9
10
1
```



```
var x = 9;
function myFunction() {
    return x * x;
}
document.write(myFunction());
x = 5;
document.write(myFunction());
```

Code 2

Global LE creation

{ outer : null, x : undefined, myFunction: fn }

Global LE execution

{ outer : null, x : 5, myFunction: fn }

myFunction LE creation / execution

{ outer : global }

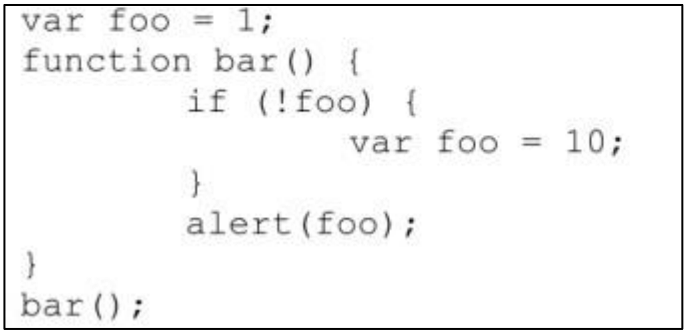
myFunction LE creation / execution:2

{ outer: global }

Output:

81

25



```
var foo = 1;
function bar() {
        if (!foo) {
                var foo = 10;
        }
        alert(foo);
}
bar();
```

Code 3

Global LE creation

{ outer : null, foo : undefined, bar : fn }

Global LE execution

{ outer : null, foo : 1 , bar : fn }

bar LE creation

{ outer : global, foo : undefined}

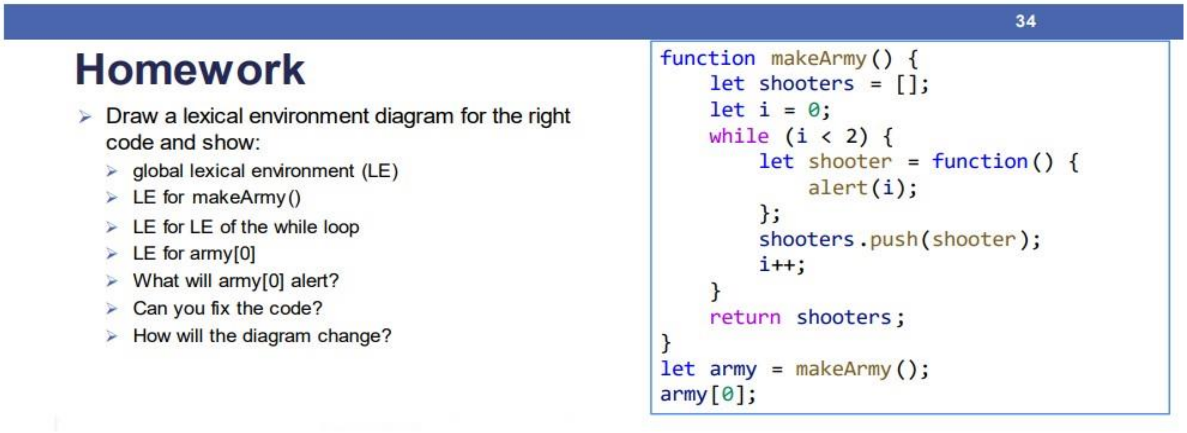
bar LE execution

{ outer : global , foo : 10 }

Output:

10

3. The following code creates an array of `shooters`. Every function is meant to output its number. But something is wrong…



34

Homework

- Draw a lexical environment diagram for the right code and show:
  - global lexical environment (LE)
  - LE for makeArmy()
  - LE for LE of the while loop
  - LE for army[0]
  - What will army[0] alert?
  - Can you fix the code?
  - How will the diagram change?

```
function makeArmy() {
    let shooters = [];
    let i = 0;
    while (i < 2) {
        let shooter = function() {
            alert(i);
        };
        shooters.push(shooter);
        i++;
    }
    return shooters;
}
let army = makeArmy();
army[0];
```

Why do all of the shooters show the same value? Fix the code so that they work as intended.

Global LE creation

{ outer : null } { TDZ : army }

Global LE execution

{ outer : null , army : 0: shooter1fn, 1: shooter2fn }

makeArmy LE creation

{ outer : null } TDZ : { shooters , i }

makeArmy LE execution

{outer: null, shooters : [0: shooter:fn , 1 : shooter:fn] , i=2 }

while LE creation:

{ outer : makeArmy }   TDZ : { shooter }

while LE execution:

{ outer : makeArmy, shooter: fn }

army[0] LE creation / execution

{ outer : global , i:2 // from closure }

Answer: Program is returning always 2 because the functions are stored without execution having closure value as 2. If we execute them now, the closure value of i is 2.

Fix:

The program can be fixed in two ways

- Using already execution way
  shooters.push(shooter());

Using bind which will return a function with fixed parameter
shooters.push(shooter.bind(this,i));