Danyal Javed (612344)

# Question 1

## 1.1: Global lexical environment (LE)

|  |  |
| --- | --- |
| function makeArmy() |  |
| let army |  |
| outer | null |

## 1.2 LE for makeArmy()

|  |  |
| --- | --- |
| let shooters | [] |
| let i | 2 |
| while (i < 2) |  |
| outer |  |

## 1.3 LE for the while loop

|  |  |
| --- | --- |
| function () { alert ( i ); }, | empty |
| function () { alert ( i ); }, | empty |
| outer |  |

## 1.4 LE for army [0]

|  |  |
| --- | --- |
| alert ( i ); | 2 |
| alert ( i ); | 2 |
| outer |  |

## 1.5 What will army [0] alert?

It will alert 2 both times.

## 1.6 Can you fix the code?

Problem is that for each iteration new lexical environment is created. So, to fix this, we can copy the value of i into a variable inside loop, like this:

function makeArmy() {

let shooters = [];

let i = 0;

while (i < 2) {

let j = i;

let shooter = function() {

alert( j );

};

shooters.push(shooter);

i++;

}

return shooters;

}

let army = makeArmy();

army[0];

## 1.7 How will the diagram change?

### LE for the while loop

|  |  |
| --- | --- |
| function () { alert ( j ); }, | empty |
| function () { alert ( j ); }, | empty |
| outer |  |

### LE for army [0]

|  |  |
| --- | --- |
| alert ( j ); | 0 |
| outer |  |

# Question 2

function printNumbers(from, to) {

let cur = from;

function doit() {

alert(cur);

if (cur == to) {

clearInterval(interval);

}

cur++;

}

doit();

let interval = setInterval(doit, 1000);

}

printNumbers(1, 5);

# Question 3

**setTimeout** will run only **after** the current code has finished.

The i will be the **last** one: 100000000.