**Lab 6**

**Question1:** Draw a lexical environment diagram for the right code and show

|  |  |  |
| --- | --- | --- |
| Lexical Environment: army(0) | | **TDZ**  LE{i=2} |
| **Lexical Environment: while**  **Loop:0**  **LE {i=0, shooters:[]}** | | **TDZ**  **Loop:0**  LE{shooter:fn} |
| **Loop:1**  **LE {i=1, shooters:[]}** | | **TDZ**  **Loop:01**  LE{shooter:fn} |
| **Functional Lexical Environment: makeArmy**  **Arguments{length:0}** | | **TDZ**  LE{shooters:[], i} |
| Outer:global | |
| **Global Lexical Environment** | | **TDZ**  LE{army:fn} |
| **Lexical Environment:army(0)**  Outer:null | **TDZ** |
| Outer:null | |

The existing code does not alert anything. Rather, it will return the function f(){alert(i);}. Also, please be noted that we are adding function inside the array rather than calling function. So, it will not alert anything

For showing alert we need to execute the function using IIFE pattern. This () indicates the self-executing of the function. On fixing this, the code will alert 2 as the last set value of i=2, where i is the free variable and on executing this function this value of I is displayed on calling the function

//existing code

// 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];

//////////////////////////////////////////////////////////////////////////////

//Solution:army[0] will not alert any thing as the army[0] is a function.

//revised code

function makeArmy() {

  let shooters = [];

  let i = 0;

  while (i < 2) {

    let shooter = function () {

      alert(i);

      return;

    };

    shooters.push(shooter);

    i++;

  }

  return shooters;

}

let army = makeArmy();

army[0].apply();

**Question 2:**

Write a function printNumbers(from, to) that outputs a number every second, starting from from and ending with to

const printNumbers = function (from, to) {

  let interval = setInterval(() => {

    while (from <= to) {

      console.log(from);

      from = from + 1;

    }

  }, 1000);

  setTimeout(function () {

    clearInterval(interval);

  }, to \* 1000);

  return;

};

printNumbers(1, 3);

**Question 3:** In the code there's a setTimeOut call scheduled, then a heavy calculation is run, that takes more than 100ms to finish.

* when will the scheduled function run?
  + The setTimeout function will run as soon as possible after completing the current code execution. Hence, the correct answer is the schedule will be called **“After the loop”**
* What is alert going to show?
  + The Alert is going show result as: 100000000
* let i=0;
* setTimeout(()=>{
* console.log(i);
* },0);
* //assume that the time to execute this function is >100ms
* for(let j=0;j<1000000000;j++){
* i++;
* }