1. Answer
   1.1. document.write(x) inside function C print ->undefined because it found the variable is declared in that block and hoisted at the top.
   1.2. document.write(a) inside function C print -> 8 because it found the local variable inside the block.
   1.3. Document.write(b) inside function f print-> 8
   1.4. Document.write(b) inside function C print-> 9
   1.5. Document.write(b) in global scope print->10
   1.6. Document.write(x) in global scope print->1

2. Answer
   2.1. The global scope is the scope which is visible to all other scopes. In javascript global scope is the window object;
   2.2. The variable which is declared inside the function and only it can be accessed inside the function and not visible to outside of it is called local scope
3. Answer
   3.1. A) No, because A is the global scope and it doesnot have visibility to access the variable inside the functional scope.
   3.2. B) Yes, because B is the functional scope and it can access all the variable which is defined in the global scope.
   3.3. C) No, because C is the scope which is inside the inner function of scope B so it doesnot have access to the that scope
   3.4. D) Yes, because A is the global scope
   3.5. E) Yes, because C is the inner function scope B so anything declared in the outer space is visible to the inner scope.
4. Answer
   4.1. First x is set to 9 and it function return 89 and **print 89**
   4.2. then x is set to 5 and function return the 25 and **print 25**
5. Answer
   5.1. Var foo is declared in the global scope. When the bar function is called, it found the variable is declared with same name in that function scope. So the variable declared in the global space with same name gets shadowed. Inside the function bar variable foo is hoisted to the top of that scope and has the value undefined. Then the if condition becomes true and foo is sets to 10 and **alert print 10**
6. **Answer**

```
var count = (function(){

    var counter = 0;
    const addCounter = function(){
        return ++counter;
    };
    const resetCounter = function(){
        counter = 0;
        return counter;
    };
    return{
        add: addCounter,
        reset: resetCounter
    };
})();
```

7. Answer
   In the above example the free variable is counter because It is not a parameter and declared inside the closure.
   Free variable are the variable which is not a parameters and declared inside in a function. These variable comes from the outer scope of that function.
8. Answer

```
var make_adder = function(inc){
    var counter = 0;
    return function(){
        counter+=inc;
        return counter;
    };
};
```

9. Answer
   Simple modification is to make the module and move all the variable declaration and list of the function into that module and give it a name.

10. Answer

```
let employee = (function(){
    let name = "";
    let age = "";
    let salary = 0.0;
    let setAge = function(newAge){
       age = newAge;
    };
    let setSalary = function(newSalary){
        salary = newSalary;
    };
    let setName = function(newName){
        name = newName;
    };
    let getAge = function(){
        return age;
    };
    let getSalary = function(){
        return salary;
    };
    let getName = function(){
       return name;
    };
    let increaseSalary = function(percentage){
        setSalary(getSalary()*(1+percentage/100));
    };
    let incrementAge = function(){
        setAge(getAge()+1);
    };
    return {
```

```
        setAge,setSalary, setName,increaseSalary,incrementAge,
        toString: function(){
            return `Name:${getName()} Age:${getAge()} Salary
             ${getSalary()}`;
        }

    };

}());
```

I have return extra toString method and make it public to print the name salary and age for the test purpose;

11. Answer

```
let address = (function () {
    let state = "";
    let zip = "";;
    let city = "";;
    let setState = function (newState) {
        state = newState;
    };
    let setZip = function (newZip) {
        zip = newZip;
    };
    let setCity = function (newCity) {
        city = newCity;
    };
    let getCity = function () {
        return city;
    };
    let getState = function () {
        return state;
    };
    let getZip = function () {
        return zip;
    }
    return {
        setCity, setState, setZip, getCity, getState, getZip,
        toString: function () {
            return `City ${getCity()} State ${getState()} Zip ${getZip()}`;
        }
    };

})();
```

```
let employee = (function () {
    let name = "";
    let age = "";
    let salary = 0.0;
    let address = null;
    let setAddress = function (newAddress) {
        address = newAddress;
    }
    let getAddress = function () {
        return address;
    }
    let setAge = function (newAge) {
        age = newAge;
    };
    let setSalary = function (newSalary) {
        salary = newSalary;
    };
    let setName = function (newName) {
        name = newName;
    };
    let getAge = function () {
        return age;
    };
    let getSalary = function () {
        return salary;
    };
    let getName = function () {
        return name;
    };
    let increaseSalary = function (percentage) {
        setSalary(getSalary() * (1 + percentage / 100));
    };
    let incrementAge = function () {
        setAge(getAge() + 1);
    };
    return {
        setAge, setSalary, setName, increaseSalary, incrementAge,
        getAddress, setAddress, address,
        toString: function () {
            return `Name:${getName()} Age:${getAge()} Salary ${getSalary()}\nA
ddress: ${getAddress().toString()}`;
        }

    };

}());
```