1. Output is: undefined -> 8 -> 8 -> 9->10->1
2. Local Scope: variables declared with a JavaScript function become Local to the function.

Global Scope: variables declared outside of a function are window global variables and are accessible everywhere in the program.

1. No Scope A do not have access to scope B and C

Yes it does

No it does not

Yes it does

Yes it does

1. 81, 25
2. 10
3. var add = (function () {
4. var counter = 0;
5. return {
6. add: function(){
7. counter+=1;
8. document.write(counter);
9. },
10. reset: function(){
11. counter = 0;
12. }
13. }
14. })();
15. add.add();
16. add.add();
17. The free variable is counter. It is defined in the outer function scope and is closed up by the inner function, closure.

Free variable is a variable that is neither locally declared nor passed as parameter.

8. Move the functions and variables inside a modulo function. Make them local

9.

var make\_adder = (function (){

    var counter = 0;

        return function(inc){

            counter+=inc;

            document.write(counter);

        }

    })();

    make\_adder(5);

    make\_adder(5);

10

var emp = function(){

    var employee ={

       'nam': 'John',

       age:40,

       salary:6000,

       getAge:function(){return this.age},

       getSalary:function(){return this.salary},

       getName:function(){return this.nam},

       setAge:function(newAge){age = newAge;},

       setSalary: function(newSalary){salary = newSalary},

       setName: function(newName){nam = newName},

       increaseSalary:function(percentage){return this.getSalary() + this.getSalary() \* percentage},

       incrementAge: function() {return  this.getAge() + 10}

    }

  function setName(name){

  return  employee.setName(name);

  }

    function setSalary(){

        return  employee.setSalary();

        }

    function setAge(age){

            return  employee.setAge(age);

            }

    function increaseSalary(p){

        return  employee.increaseSalary(p);

    }

    function incrementAge(){

        return  employee.incrementAge();

    }

       return {

                setName: setName,

                setSalary: setSalary,

                setAge: setAge,

                increaseSalary:increaseSalary,

                incrementAge:incrementAge

       }

}();

console.log(emp.increaseSalary(2));

console.log(emp.incrementAge());

11.

var emp = function(){

    var employee ={

       'nam': 'John',

       age:40,

       salary:6000,

       getAge:function(){return this.age},

       getSalary:function(){return this.salary},

       getName:function(){return this.nam},

       setAge:function(newAge){age = newAge;},

       setSalary: function(newSalary){salary = newSalary},

       setName: function(newName){nam = newName},

       increaseSalary:function(percentage){return this.getSalary() + this.getSalary() \* percentage},

       incrementAge: function() {return  this.getAge() + 10}

    }

  function setName(name){

  return  employee.setName(name);

  }

    function setSalary(){

        return  employee.setSalary();

        }

    function setAge(age){

            return  employee.setAge(age);

            }

    function increaseSalary(p){

        return  employee.increaseSalary(p);

    }

    function incrementAge(){

        return  employee.incrementAge();

    }

       return {

                setName:  function(n){return setName(n)},

                setSalary: function(s){return setSalary(s)},

                setAge: function(a){return setAge(a)},

                increaseSalary:function(n){return increaseSalary(n)},

                incrementAge:function(){return incrementAge()},

       }

}();

console.log(emp.increaseSalary(2));

console.log(emp.incrementAge());

12.

var emp = function(){

    var employee ={

       'nam': 'John',

       age:40,

       salary:6000,

       getAge:function(){return this.age},

       getSalary:function(){return this.salary},

       getName:function(){return this.nam},

       setAge:function(newAge){age = newAge;},

       setSalary: function(newSalary){salary = newSalary},

       setName: function(newName){nam = newName},

    }

    function getSalary(){

        return  employee.getSalary();

        }

    function getAge(){

            return  employee.getAge();

            }

    employee.increaseSalary = function(percentage){return getSalary() +getSalary() \* percentage}

    employee.incrementAge = function() {return  getAge() + 10} ;

       return employee

}();

console.log(emp.increaseSalary(2));

console.log(emp.incrementAge());

13.

    var emp2= (function(){

        emp.extension = function () {

            var address;

           emp.setAddres = function setAddress(newAddress){

                address = newAddress;

            }

            emp.getAddres = function getAddress(){

                return address;

            }

            };

            return emp;

        })(emp || {});

    console.log(emp2);

14) Error: Hattori

15) Success: Hattori

Error: Yoshi

16) Success

error

Error caught