Nahom Berta 611108
Lab 7/8

1) undefined889101
   undefined: x is hoisted, 8: value of parameter a, 8: function f's parameter b is set equal to a which is 8 when called, 9: b is still 9 outside function f, 10: global scope b, 1: global x
2) Global scope refers to a value that is visible from anywhere; in the case of JavaScript, it is a variable declared outside any function or block. Local scope refers to a value that is visible only to its enclosing environment such as a block or a function, and is inaccessible from outside that environment.
3) a) No, statements from scope A cannot access any values declared within function scopes B and C.
   b) Yes; they would be accessing free variables along the scope chain.
   c) No: similar reason as in a.
   d) Yes; they would be accessing free variables along the scope chain.
   e) Yes; they would be accessing free variables along the scope chain.
4) 8125: writes the squares of 9 and 5
5) 10: foo is initialized to 10 because the if clause runs given that foo is undefined then and !undefined is truthy.
6)
```
const count = {  "counter": 0,
      add : function() {
                this.counter += 1;
       },
      reset: function() {
                this.counter = 0;
      }
};
```
7) counter is a free variable. A free variable is a variable being accessed by a function though that variable is outside its scope.
8)
```
const make_adder = function(inc) {
          let counter = 0;
          return function() {
```

```
            counter += inc;
        }
};
```

9) Put whole page inside function braces and invoke it, i.e. (//whole page)(). ALternatively, window.onload can be set to a function holding the whole page, i.e window.onload = function() { //whole page };

10)
```
const employee = (function() {
        //private fields
        let name;
        let age;
        let salary;

        //public methods
        function setAge(newAge) { this.age = newAge; }
        function setSalary(newSalary) {
            this.age = newSalary; }
        function setName(newName) {
            this.name = newName; }
        function incrementAge() {
            this.age = getAge() + 1; }
        function incrementSalary(percentage) {
            this.salary = getSalary() *
            (1 + percentage); }

        //private methods
        function getSalary() { return this.salary; }
        function getAge() { return this.age; }
        function getName() { return this.name; }

        return {
            setAge: setAge,
            setName: setName,
            setSalary: setSalary,
            incrementSalary: incrementSalary,
            incrementAge: incrementAge
        };
    })();
```

```
11)  employee.address = undefined;
     employee.setAddress = function(newAddress) { this.address =
newAddress; }
     employee.getAddress = function() { return this.address; }
```