Lesson 6: Scope, closures and encapsulation lab Solutions

1) Filter banned word: Soln 1

```
function filterString(str,banned)
{
        var strArray=str.split(" ");
        var allowedStr=[];

        for(let i=0;i<strArray.length;i++)
        {
        if(strArray[i]!== banned)
    {
                allowedStr.push(strArray[i]);
    }
}
console.log(allowedStr);
return allowedStr.join(" ");
}

// execution
let allowed=filterString("I do not like alchol!",'not');
console.log(allowed);
// Output: I do like alchol!

// Exercise- 1 :  Filter banned word: Soln 2
```

```
function filterBannedWord(str,banned)
{
        return str.split(" ").filter(word=>word!==banned).join(" ");
}
// Excution
const str='I am not going to nightclub. not always!';
console.log(filterBannedWord(str,'not'));
 //Output: I am going to nightclub. always!
```

2) Bubble sort

```
//Exercise-2:  Buuble sort
function bubbleSort(items) {
    var length = items.length;

    for (let i = 0; i < length; i++) {
        for (let j = 0; j < (length - i - 1); j++) {

            if(items[j] > items[j+1]) {
                //Swap the numbers
                let tmp = items[j];
                items[j] = items[j+1];
                items[j+1] = tmp;
            }
        }
```

```
    }
        return items;
}
// Execution:
console.log(bubbleSort([6,4,0,3,-2,1]));
//output: [-2,0,1,3,4,6]
```

3) Teacher/Person inheritance

```
function Person(fname,lname){
        this.firstName=fname;
    this.lastName=lname;

}

function Teacher(fname,lname)
{
        Person.call(this,fname,lname);
        this.teach=function(subject){
         return `${this.firstName} is now teaching ${subject}`;
        };
```

```
}

var teacher=new Teacher('John','Doe');
//document.getElementById("demo5").innerHTML=teacher.teach('Database
');
console.log(teacher.teach('Database'));
// Output: John is now teaching Database



//Exercise-3: Teacher/Person inheritance using Object.create()
let person = {
    firstName: '',
    lastName: '',
    getName(name) {
        return this.firstName=name ;
    }
};

function Teacher(fname)
{
        this.teach=function(subject){
                return `${this.getName(fname)} is now teaching ${subject} `;
        };

}
Teacher.prototype=Object.create(person);
```

```
//Execution:
var teacher=new Teacher('John');
console.log(teacher.teach('Database'););
// Output: John is now teaching Database
```

4) Person/Student/Professor Inheritance

// Exercise -4 : Person/Student/Professor Inheritance using Class

```
class Person{
        constructor(name,age){
      this.name=name;
      this.age=age;
    }

    greeting()
    {
        console.log(`Greetings, my name is <em
style='color:blue'>${this.name}</em> and I am ${this.age} years old.`);
    }

    salute(){
        console.log(`Good Morning! and in case I dont see you, good
afternoon, good evening and good night!`)
    }
}
```

```
// creating Student Object
class Student extends Person{
        constructor(name,age,major){
      super(name,age);
      this.major=major;
    }

     //output to console
    greeting(){
        console.log(`Hey, my name is ${this.name} and I am studying
${this.major}`);
    }

}



// Creating Professor Object
class Professor extends Person{
        constructor(name,age,department){

    super(name,age);
    this.department=department;
```

```
    }

    //output to console
    greeting(){
    console.log(`Good day, my name  is ${this.name} and I am in the ${this.department} department.`);
    }
}

// create Professor and Student Objects
let professor=new Professor('John',40,'Economics');
        professor.greeting();
    professor.salute();

let student=new Student('Amanuel',30,'Computer Science');
        student.greeting();
    student.salute();



// Output:

/*  Good day, my name  is John and I am in the Economics department.
        Good Morning! and in case I dont see you, good afternoon, good evening and good night!

  Hey, my name is Steve and I am studying Computer Science.
```

```
  Good Morning! and in case I dont see you, good afternoon, good evening and good night!
*/



/** Exercise -4 : Person/Student/Professor Inheritance
  Using Object Prototype approach
 **/
//1 person Object
var person={
        name:'unknown',
    age:0,
    greeting:function(){
        console.log(`Greetings, my name is ${this.name} and I am ${this.age} years old.`);
      },


    salute: function(){
        console.log(`Good Morning! and in case I dont see you, good afternoon, good evening and good night!`);
      }


};
```

```
//2 student object ( Being inhereted from person object )
var student=Object.create(person);
   student.major='Computer Science';
   student.name='Amanuel';

   //override greeting function of person object
   student.greeting=function(){
   console.log(`Hey, my name is ${this.name} and I am studying
${this.major}.`);
   };

   console.log(student.greeting());
   console.log(student.salute());



//3 professor object (Being inhereted from person object
var professor=Object.create(person);
professor.name='Mark';
professor.department='Human Resource';

        //Overriding greeting method of person object
professor.greeting=function(){
```

```
        console.log(`Good day, my name  is ${this.name}  and I am in the ${this.department} department.`);
};


//execution
  console.log(professor.greeting());
  console.log(professor.salute());



// Output
/*
Hey, my name is Amanuel and I am studying Computer Science.
Good Morning! and in case I dont see you, good afternoon, good evening and good night!

 Good day, my name  is Mark  and I am in the Human Resource department.
 Good Morning! and in case I dont see you, good afternoon, good evening and good night!
*/
```

```
/** Exercise -4 : Person/Student/Professor Inheritance
  Using Object Function constructor approach
 **/
 //1 Person
function Person(name, age) {
  this.name = name;
  this.age = age;

}
Person.prototype.greeting=function(){
        console.log(`Greetings, my name is ${this.name} and I am ${this.age} years old.`);

};

Person.prototype.salute=function(){
                console.log(`Good Morning! and in case I dont see you, good afternoon, good evening and good night!`);

};

//2 Student
function Student(name,age,major) {
  Person.call(this,name,age);
  this.major = major;
```

```
}

Student.prototype.greeting=function(){
        console.log(`Hey, my name is ${this.name} and I am studying ${this.major}`);
    return `Hey, my name is ${this.name} and I am studying ${this.major}`;
};



//3. Professor

function Professor(name,age,department) {
        Person.call(this,name,age);
        this.department = department;
}
Professor.prototype.greeting=function(){
        console.log(`Good day, my name  is ${this.name}  and I am in the ${this.department} department.`);

    return `Good day, my name  is ${this.name}  and I am in the ${this.department} department.`
};



// Execution:
```

```
// Inherit Student.prototype from Person.prototype
Student.prototype=Object.create(Person.prototype);

let student=new Student('Steve',26,'Economics');
        student.greeting();
         student.salute();

// Inherit Professor.prototype from Person.prototype
Professor.prototype=Object.create(Person.prototype);
let professor=new Professor('Selam',40,'Computer Science');
        professor.greeting();
        professor.salute();

// OutPut:
/*
Greetings, my name is Steve and I am 26 years old.
Good Morning! and in case I dont see you, good afternoon, good evening and good night!

Greetings, my name is Selam and I am 40 years old.
Good Morning! and in case I dont see you, good afternoon, good evening and good night!
*/
```

Thank you.

Amanuel k Gebru