Module 3

Exercise 5 (ES6 Part-2)

1. Filter unique array members using Set.

A1.) Js

```
//Q1 Filter unique array members using Set.
let filterArray=(arr)=>{
    if(!arr.length || arr.length===1) return arr;
    let s=new Set();

    arr.forEach(element => {
        s.add(element);
    });

    return [...s];
}


console.log(filterArray([1,2,5,2,3,4,8,5,1]));
```

Output

```
▼ (6) [1, 2, 5, 3, 4, 8] i
    0: 1
    1: 2
    2: 5
    3: 3
    4: 4
    5: 8
    length: 6
  ▶ [[Prototype]]: Array(0)
```

2. Filter anagrams using Map.

A2.) JS

```
//Q2 Filter anagrams using Map.
let anagrams = (str1, str2) => {
    if (str1.length !== str2.length) return false;

    str1 = str1.toLowerCase();
    str2 = str2.toLowerCase();

    const charMap = new Map();

    for (let char of str1) {
        charMap.set(char, (charMap.get(char) || 0) + 1);
    }

    for (let char of str2) {
        if (!charMap.has(char)) return false;
        charMap.set(char, charMap.get(char) - 1);
        if (charMap.get(char) < 0) return false;
    }

    return true;
};


console.log(anagrams("Listen","Silett"));
console.log(anagrams("Listen","Silent"));
```

Output

3. Write a program to implement inheritance upto 3 classes.The Class must contain private and public variables and static functions.

A3.) JS

```
/*Q3 Write a program to implement inheritance upto 3 classes.
The Class must contain private and public variables and static functions.*/

class Person{
    firstname;
    #address;
    constructor(name,address){
        this.firstname=name;
        this.#address=address;
    }

    static getInfo(person){
        console.log(person.firstname);
    }
}

class Employee extends Person{
    designation;
    #salary;

    constructor(designation,salary,name){
        super(name);
        this.designation=designation;
        this.#salary=salary;
    }

    static info(employee){
        console.log(`${employee.firstname} is a ${employee.designation} his salary is ${employee.#sal
    }
}
```

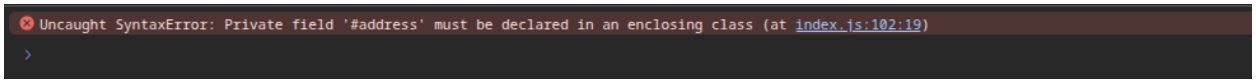
```
class Developer extends Employee{
    canCode;
    constructor(canCode,designation,name){
        super(designation,15000,name);
        this.canCode=canCode;
    }

    static ask(developer){
        console.log(`${developer.firstname} as a ${developer.designation} can code ${developer.canCod
    }
}


const developer=new Developer(true,"Software Deveolper Trainee","Dhruv");
const person=new Person("Dhruv","Delhi");
console.log(person.#address);

Developer.ask(developer);
Developer.info(developer);
Developer.getInfo(developer);
```

Output



Error encountered on accessing private variables through object

```
Dhruv as a Software Deveolper Trainee can code true
Dhruv is a Software Deveolper Trainee his salary is 15000
Dhruv
```

Static function are only accessible by Class itself not by its instance and static function can only access static variable and can accessed class variable if we provide instance as a parameter

4. Write a program to implement a class having static functions

A4.) JS

```js
//Q4 Write a program to implement a class having static functions
class Utils {

  static add(a, b) {
    return a + b;
  }

  static multiply(a, b) {
    return a * b;
  }
  instanceMethod() {
    return "This is an instance method.";
  }
}


const sum = Utils.add(5, 10);
console.log(`The sum is: ${sum}`);

const product = Utils.multiply(3, 4);
console.log(`The product is: ${product}`);
const utilsInstance = new Utils();
try {
  utilsInstance.add(1, 2);
} catch (error) {
  console.log(`Error calling static method on an instance: ${error.message}`);
}
console.log(utilsInstance.instanceMethod());
```

Output

```
The sum is: 15
The product is: 12
Error calling static method on an instance: utilsInstance.add is not a function
This is an instance method.
```

5. Import a module containing the constants and method for calculating area of circle, rectangle, cylinder.

A5.) JS

```
//Q5 Import a module containing the constants and method for calculating area of circle, rectangle, c
//All this are present in area.js
console.log(PI);
console.log(areaOfCircle(5));
console.log(areaOfRectangle(3,6));
let [curvedSurfacearea,totalSurfaceArea]=areaOfCylinder(5,12);

console.log(`The Curved Surface area of the Cylinder is ${curvedSurfacearea}`);
console.log(`The Total Surface area of Cylinder is ${totalSurfaceArea}`);
```

area.js

```
export const PI=3.14;

export function areaOfCircle(radius){
    let area=PI*Math.pow(radius,2)
    return area;
}

export function areaOfRectangle(length,breadth){
    let area = length*breadth;
    return area;
}

export function areaOfCylinder(radius,height){
    let curvedSurfaceArea=2*PI*radius*height;
    let totalSurfaceArea=curvedSurfaceArea+(2*PI*Math.pow(radius,2));
    return [curvedSurfaceArea,totalSurfaceArea];
}
```

Output

```
3.14
78.5
18
The Curved Surface area of the Cylinder is 376.8
The Total Surface area of Cylinder is 533.8
```

6. Import a module for filtering unique elements in an array.

A6.) JS

```
//Q6 Import a module for filtering unique elements in an array.
//This two functions are present inside filter.js
console.log(`Filter with set ${filterWithSet([1,2,5,2,4,7,4,8])}`);

console.log(`Filter without set ${filterWithoutSet([1,2,4,5,8,3,5,7,2,4])}`);
```

filter.js

```
export function filterWithSet(arr){
    let s=new Set();
    arr.forEach(element => {
        s.add(element);
    });

    return [...s];
}

export function filterWithoutSet(arr){
    return arr.filter((e,i,self)=> i===self.indexOf(e));
}
```

Output

```
Filter with set 1,2,5,4,7,8
Filter without set 1,2,4,5,8,3,7
```

7. Write a program to flatten a nested array to single level using arrow functions.

A7.) JS

```js
//Q7 Write a program to flatten a nested array to single level using arrow functions.
let flattenWithFlat=(arr)=>{
    return arr.flat('Infinity');
}

let flattenWithoutFlat=(arr)=>{
    let flatArr=[];
    arr.forEach(element => {
        if(Array.isArray(element)){
            flatArr=flatArr.concat(flattenWithoutFlat(element));
        }
        else{
            flatArr.push(element)
        }
    });

    return flatArr;
}


console.log(flattenWithFlat([1,2,[3,4,[5,6,7,[8],[9,10]]]]));
console.log(flattenWithoutFlat([1,2,[3,4,[5,6,7,[8],[9,10]]]]));
```

Output

```
▸ (10) [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
▸ (10) [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```