**ES6 Exercise**

**P.s : please copy the code as it is to the console to verify them.**

**1. Filter unique array members using Set.**

**Solution :** var arr = [1,2,3,4,1,7,7,8];

var set = new Set(arr);

console.log(set);  
  
**3. Write a program to implement inheritance upto 3 classes.The Class must public variables and static functions.**

**Solution :**

class parent

{

constructor(parentId)

{

this.parentId = parentId;

}

parentfunc()

{

console.log("from parent");

}

static testfunc()

{

console.log("testfunc runs");

}

}

class child extends parent

{

constructor(childId , parentId)

{

super(parentId);

this.childId = childId;

}

}

var ch = new child(2,1);

ch.parentfunc();

parent.testfunc();

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

**Solution** :

class parent

{

constructor(parentId)

{

this.parentId = parentId;

}

static testfunc()

{

console.log("testfunc runs");

}

}

var p = new parent(2);

parent.testfunc();

**5. Import a module containing the constants and method for calculating area of circle, rectangle, cylinder.**(5 and 6 together)

**6. Import a module for filtering unique elements in an array.**(5 and 6 together in 1 answer)

Solution :

function circle(radius,pie)

{

return(pie\*radius\*radius);

}

function cylinder(radius,height,pie)

{

return (2\*pie\*radius\*(radius + height));

}

function rectangle(length, width)

{

return(length\*width);

}

function filter(arr)

{

var set = [...new Set(arr)];

return set;

P.S : using actual import and export keywords required node.js and we were told by Session owner to answer these questions like this way for this exercise(like putting js functions in separate js file only)

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

**Solution :**

var arr = [1,2,3,[4,5,6],7,8,[9,[10,11]],12];

function flatten(arr)

{

return arr.reduce((acc, val) => Array.isArray(val) ? acc.concat(flatten(val)) : acc.concat(val), []);

}

console.log(flatten(arr));

**8. Implement a linked list in es6 and implement addFirst() addLast(), length(), getFirst(), getLast().**

**Solution :**class LinkedList {

constructor() {

this.head = null;

this.tail = null;

this.count = 0;

}

get length() {

return this.count;

}

addLast(data) {

const node = {

data: data,

next: null

}

if(this.count === 0) {

this.head = node;

} else {

this.tail.next = node;

}

this.tail = node;

this.count++;

}

addFirst(data) {

const node = {

data: data,

next: null

}

const temp = this.head;

this.head = node;

this.head.next = temp;

this.count++;

if(this.count === 1) {

this.tail = this.head;

}

}

removeFirst(data) {

if(this.count > 0) {

this.head = this.head.next;

this.count--;

if(this.count === 0) {

this.tail = null;

}

}

}

getLast()

{

return this.tail;

}

getFirst()

{

return this.head;

}

removeLast(data) {

if(this.count > 0) {

if(this.count === 1) {

this.head = null;

this.tail = null;

} else {

let current = this.head;

while(current.next !== this.tail) {

current = current.next;

}

current.next = null;

this.tail = current;

}

this.count--;

}

}

}

var obj = new LinkedList();

obj.addFirst(2);

obj.addFirst(1);

obj.addLast(3);

console.log(obj.getFirst() , obj.getLast());

**9. Implement Map and Set using Es6?  
Solution :**

var map = new Map();

map.set(1,"first");

map.set(2,"second");

map.set(3,"third");

map.set(4,"fourth");

Map(4) {1 => "first", 2 => "second", 3 => "third", 4 => "fourth"}

map.get(3);

"Third"

var arr = [1,2,3,4,1,1,1,2,5,6,7,8];

var set = new Set(arr);

console.log(set);

set.add(9);

set.add(10);

console.log(set.has(5));

console.log(set.has(9));

console.log(set);

|  |  |  |
| --- | --- | --- |
| **10. Implementation of stack ?**  **Solution :**  class Stack {  constructor() {  this.items = [];  this.count = 0;  }    length() {  return this.count;  }    push(item) {  this.items.push(item);  this.count = this.count + 1;  }    pop() {  if(this.count > 0) {  this.count = this.count - 1;  }    return this.items.pop();  }    peek() {  return this.items.slice(-1);  }  }  var st = new Stack();  st.push(1);  st.push(2);  st.push(3);  st.push(4);  st.push(5);  st.push(6);  console.log("length of stack is " + st.length());  console.log("last element is " + st.peek());  st.pop();  console.log("last element is " + st.peek()); | | |