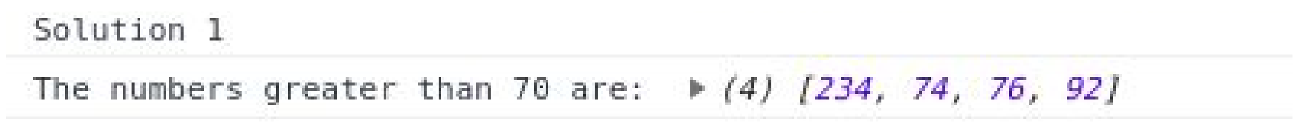
JavaScript ES6 Assignment

Q1. Given this array: `[3,62,234,7,23,74,23,76,92]`, Using the arrow function, create an array of the numbers greater than `70`.

Solution:

```
let a = [3,62,234,7,23,74,23,76,92]
const numbers = a.filter((num) => num > 70);
console.log("Solution 1");
console.log("The numbers greater than 70 are: ",numbers);
```

Output:



```
Solution 1
The numbers greater than 70 are:  ▶ (4) [234, 74, 76, 92]
```

Q2.

a. Select all the list items on the page and convert to array.

b. Filter for only the elements that contain the word 'flexbox'

c. Map down to a list of time strings

d. Map to an array of seconds

e. Reduce to get total using .filter and .map

Solution:

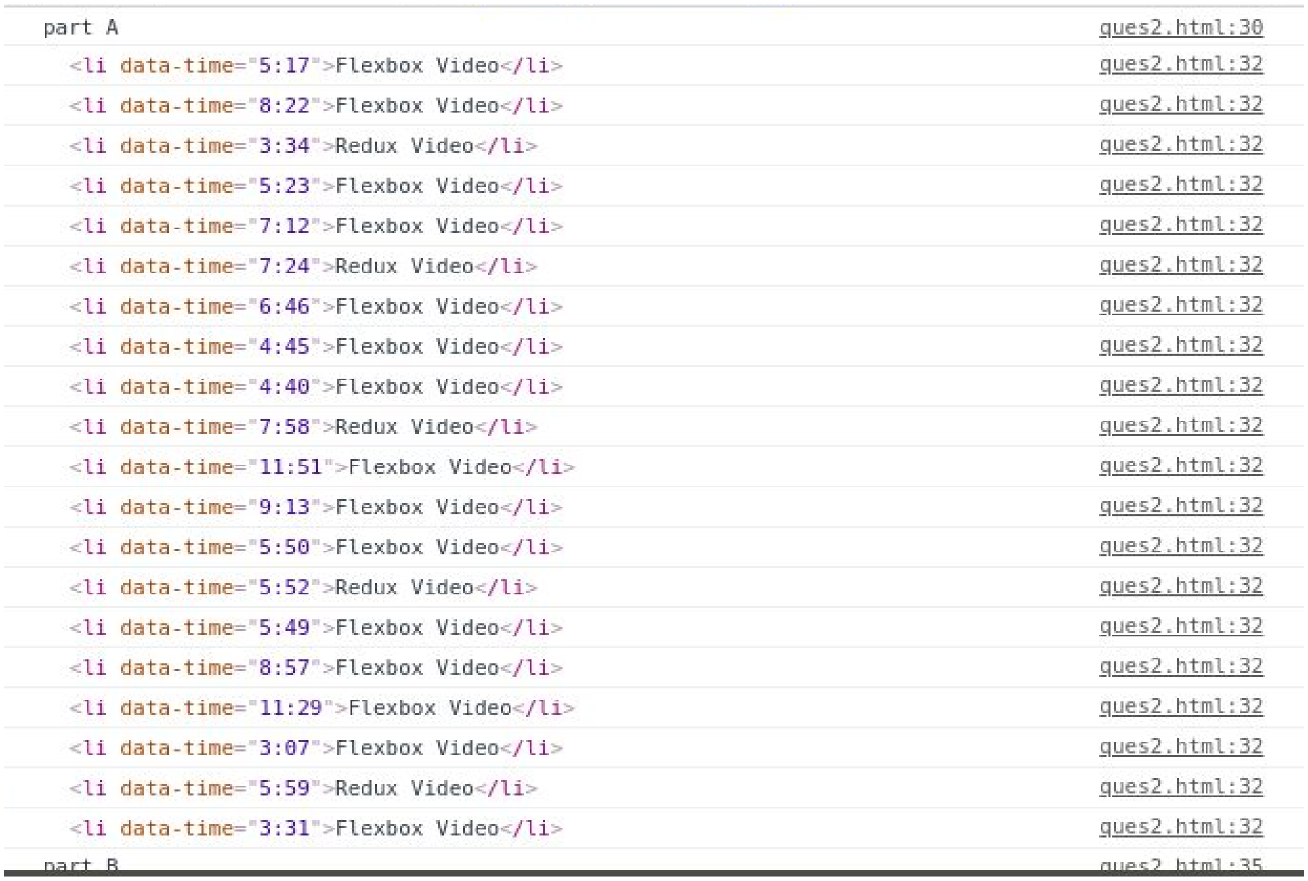
```
<html>
  <body>
     <ul>
        <li data-time="5:17">Flexbox Video</li>
        <li data-time="8:22">Flexbox Video</li>
        <li data-time="3:34">Redux Video</li>
        <li data-time="5:23">Flexbox Video</li>
        <li data-time="7:12">Flexbox Video</li>
        <li data-time="7:24">Redux Video</li>
        <li data-time="6:46">Flexbox Video</li>
        <li data-time="4:45">Flexbox Video</li>
        <li data-time="4:40">Flexbox Video</li>
        <li data-time="7:58">Redux Video</li>
        <li data-time="11:51">Flexbox Video</li>
        <li data-time="9:13">Flexbox Video</li>
        <li data-time="5:50">Flexbox Video</li>
```

```
    <li data-time="5:52">Redux Video</li>
    <li data-time="5:49">Flexbox Video</li>
    <li data-time="8:57">Flexbox Video</li>
    <li data-time="11:29">Flexbox Video</li>
    <li data-time="3:07">Flexbox Video</li>
    <li data-time="5:59">Redux Video</li>
    <li data-time="3:31">Flexbox Video</li>
  </ul>

<script>
        let ListItems = document.querySelectorAll('li');
        let itemsArray = Array.from(ListItems);

        console.log('part A');
        for(let elements of itemsArray){
            console.log(elements);
        }

        console.log('part B')
        let filteredElements = itemsArray.filter((e)=>e.innerHTML.includes('Flexbox'));
        for(let elements of filteredElements){
          console.log(elements);
        }

        console.log('part C');
        let mapItems = itemsArray.map((e)=>e.dataset.time);
        console.log(mapItems);


        console.log('part D');
        let secondsArray = mapItems.map(e=>
        {
            const y = e.split(':').map((e)=>parseInt(e));

            return (y[0]*60+y[1]);

      })

      console.log(secondsArray);
      console.log('part E');
      const reducer= (acc,current)=>acc+current;
      console.log(secondsArray.reduce(reducer));
</script>
```

```
  </body>
</html>
```

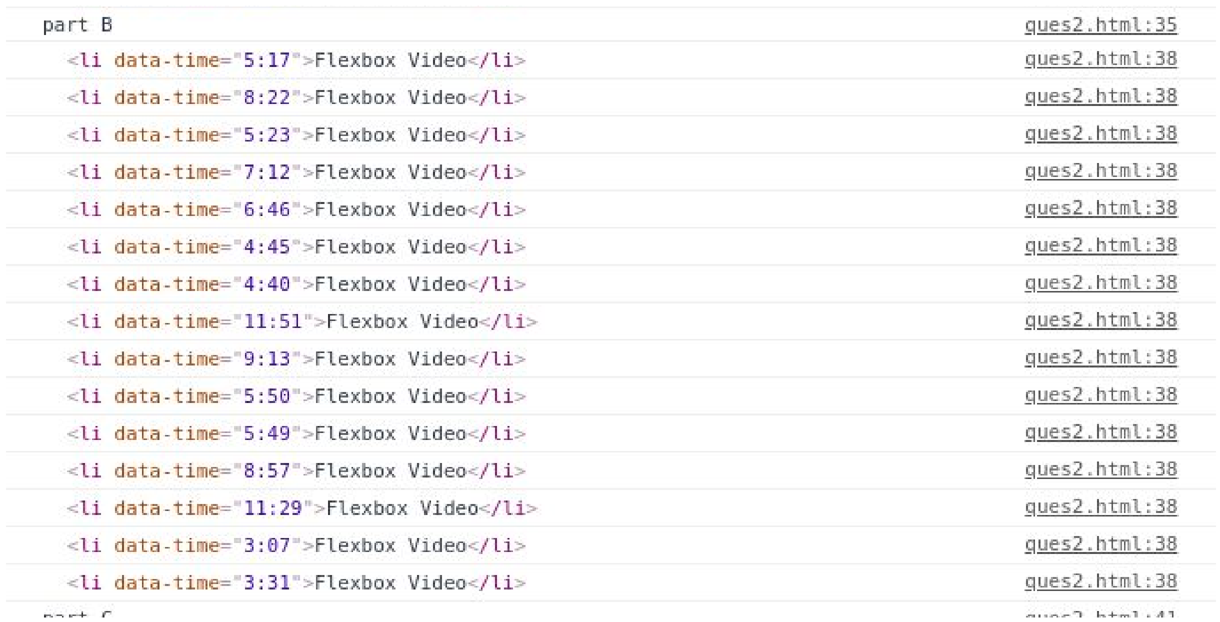
Output:

Part A:



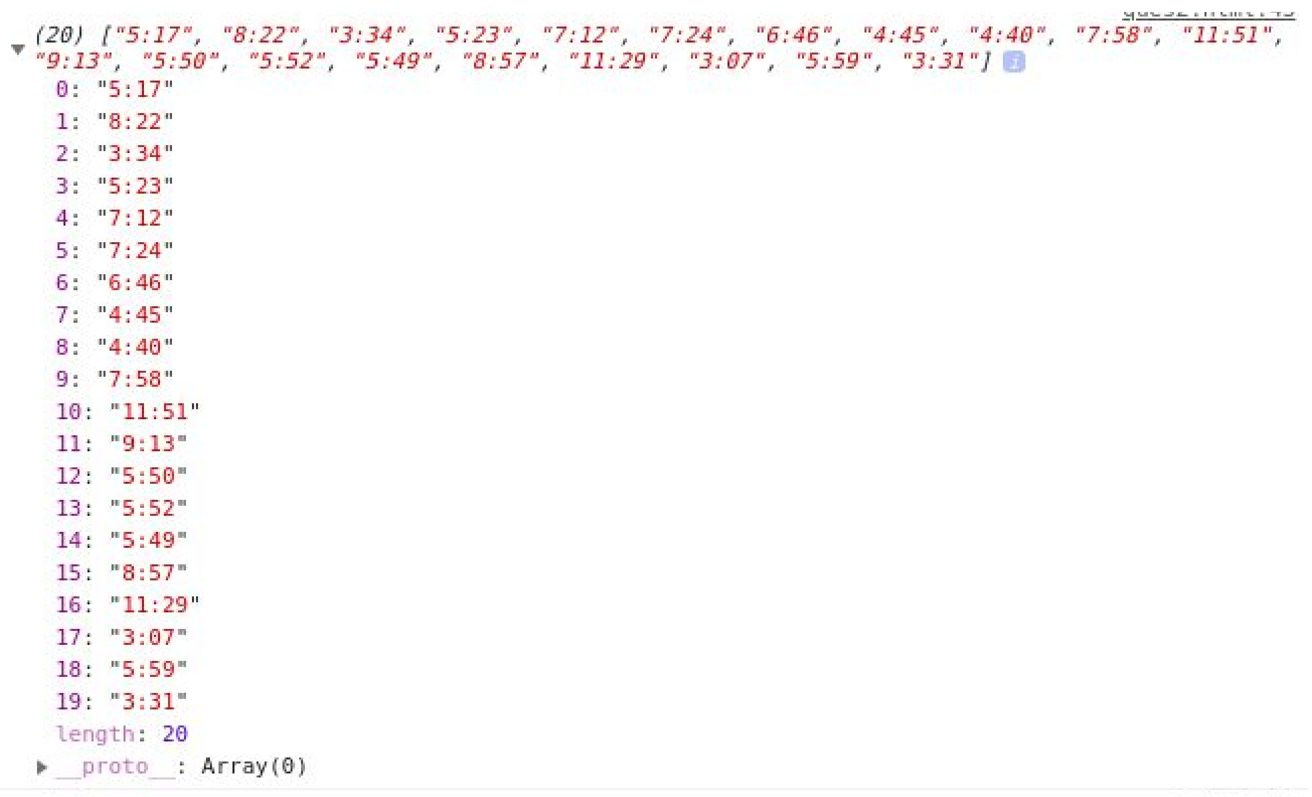
```
part A                                                        ques2.html:30
  <li data-time="5:17">Flexbox Video</li>                     ques2.html:32
  <li data-time="8:22">Flexbox Video</li>                     ques2.html:32
  <li data-time="3:34">Redux Video</li>                       ques2.html:32
  <li data-time="5:23">Flexbox Video</li>                     ques2.html:32
  <li data-time="7:12">Flexbox Video</li>                     ques2.html:32
  <li data-time="7:24">Redux Video</li>                       ques2.html:32
  <li data-time="6:46">Flexbox Video</li>                     ques2.html:32
  <li data-time="4:45">Flexbox Video</li>                     ques2.html:32
  <li data-time="4:40">Flexbox Video</li>                     ques2.html:32
  <li data-time="7:58">Redux Video</li>                       ques2.html:32
  <li data-time="11:51">Flexbox Video</li>                    ques2.html:32
  <li data-time="9:13">Flexbox Video</li>                     ques2.html:32
  <li data-time="5:50">Flexbox Video</li>                     ques2.html:32
  <li data-time="5:52">Redux Video</li>                       ques2.html:32
  <li data-time="5:49">Flexbox Video</li>                     ques2.html:32
  <li data-time="8:57">Flexbox Video</li>                     ques2.html:32
  <li data-time="11:29">Flexbox Video</li>                    ques2.html:32
  <li data-time="3:07">Flexbox Video</li>                     ques2.html:32
  <li data-time="5:59">Redux Video</li>                       ques2.html:32
  <li data-time="3:31">Flexbox Video</li>                     ques2.html:32
part B                                                        ques2.html:35
```

Part B:



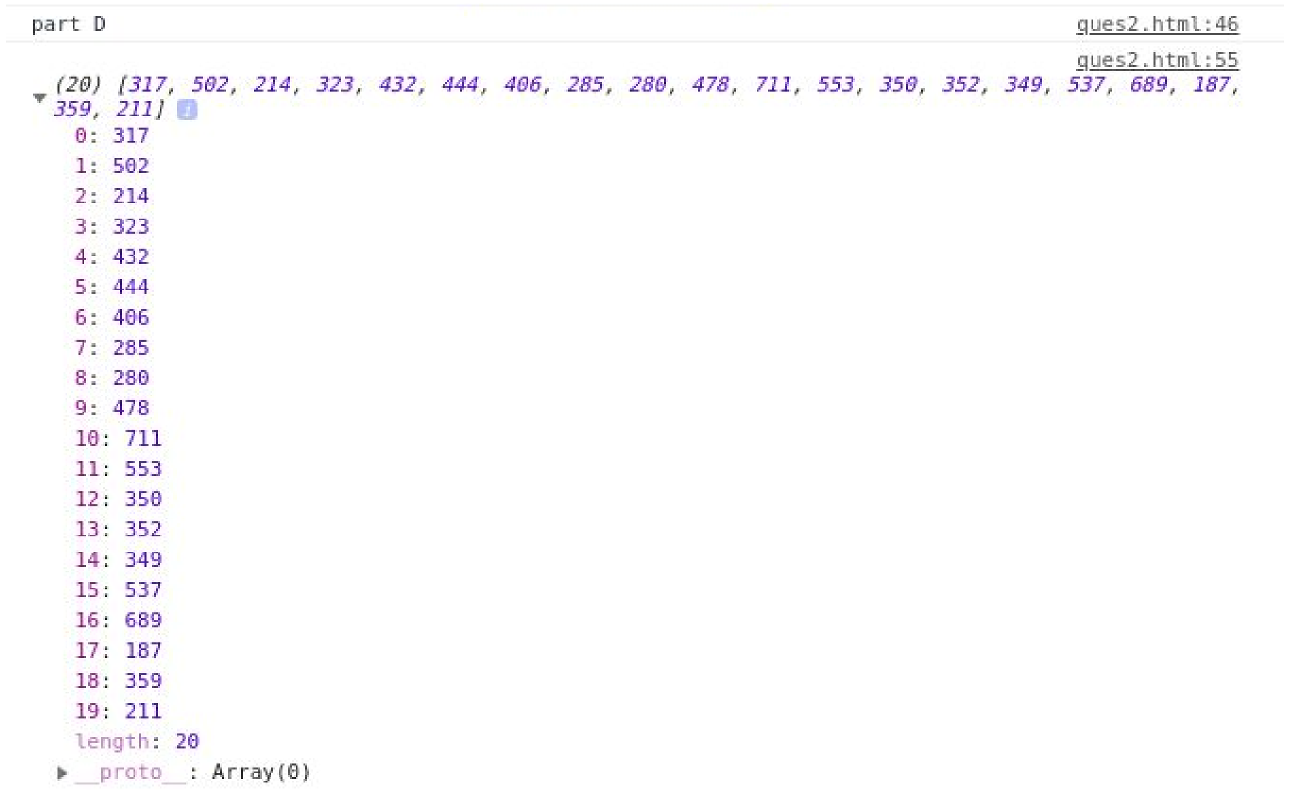
```
part B                                                        ques2.html:35
  <li data-time="5:17">Flexbox Video</li>                     ques2.html:38
  <li data-time="8:22">Flexbox Video</li>                     ques2.html:38
  <li data-time="5:23">Flexbox Video</li>                     ques2.html:38
  <li data-time="7:12">Flexbox Video</li>                     ques2.html:38
  <li data-time="6:46">Flexbox Video</li>                     ques2.html:38
  <li data-time="4:45">Flexbox Video</li>                     ques2.html:38
  <li data-time="4:40">Flexbox Video</li>                     ques2.html:38
  <li data-time="11:51">Flexbox Video</li>                    ques2.html:38
  <li data-time="9:13">Flexbox Video</li>                     ques2.html:38
  <li data-time="5:50">Flexbox Video</li>                     ques2.html:38
  <li data-time="5:49">Flexbox Video</li>                     ques2.html:38
  <li data-time="8:57">Flexbox Video</li>                     ques2.html:38
  <li data-time="11:29">Flexbox Video</li>                    ques2.html:38
  <li data-time="3:07">Flexbox Video</li>                     ques2.html:38
  <li data-time="3:31">Flexbox Video</li>                     ques2.html:38
```

Part C:



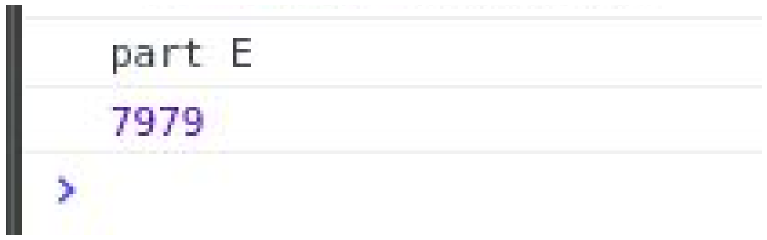
```
▾ (20) ["5:17", "8:22", "3:34", "5:23", "7:12", "7:24", "6:46", "4:45", "4:40", "7:58", "11:51",
  "9:13", "5:50", "5:52", "5:49", "8:57", "11:29", "3:07", "5:59", "3:31"]
    0: "5:17"
    1: "8:22"
    2: "3:34"
    3: "5:23"
    4: "7:12"
    5: "7:24"
    6: "6:46"
    7: "4:45"
    8: "4:40"
    9: "7:58"
    10: "11:51"
    11: "9:13"
    12: "5:50"
    13: "5:52"
    14: "5:49"
    15: "8:57"
    16: "11:29"
    17: "3:07"
    18: "5:59"
    19: "3:31"
    length: 20
  ▸ __proto__: Array(0)
```

Part D:



```
part D                                                                    ques2.html:46
                                                                          ques2.html:55
▼(20) [317, 502, 214, 323, 432, 444, 406, 285, 280, 478, 711, 553, 350, 352, 349, 537, 689, 187,
 359, 211]
   0: 317
   1: 502
   2: 214
   3: 323
   4: 432
   5: 444
   6: 406
   7: 285
   8: 280
   9: 478
   10: 711
   11: 553
   12: 350
   13: 352
   14: 349
   15: 537
   16: 689
   17: 187
   18: 359
   19: 211
   length: 20
  ▶__proto__: Array(0)
```

Part E:



```
part E
7979
>
```

Q3. Create a markup template using string literal

const song = {

name: 'Dying to live',

artist: 'Tupac',

featuring: 'Biggie Smalls'

};

Result:

"<div class="song">

<p>

Dying to live — Tupac

(Featuring Biggie Smalls)

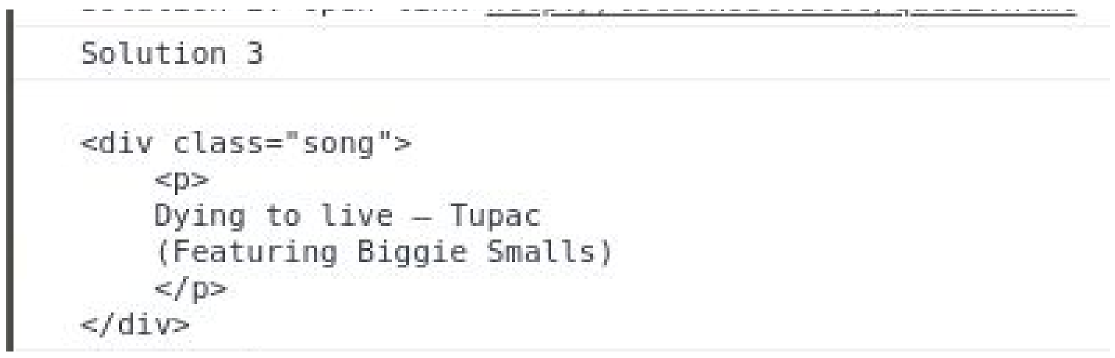
</p>

</div>“

Solution:

```
const song = {
  name: 'Dying to live',
  artist: 'Tupac',
  featuring: 'Biggie Smalls'
};

const markup = `
<div class="song">
  <p>
  ${song.name} — ${song.artist}
  (Featuring ${song.featuring})
  </p>
</div>
`;
console.log("Solution 3");
console.log(markup);
```

Output:



```
Solution 3

<div class="song">
    <p>
    Dying to live — Tupac
    (Featuring Biggie Smalls)
    </p>
</div>
```

Q4. Extract all keys inside address object from user object using destructuring ?

const user = {

firstName: ‘Sahil’,

lastName: ‘Dua’,

Address: {

Line1: ‘address line 1’,

Line2: ‘address line 2’,

State: ‘Delhi’,

Pin: 110085,

Country: ‘India’,

City: ‘New Delhi’,

},

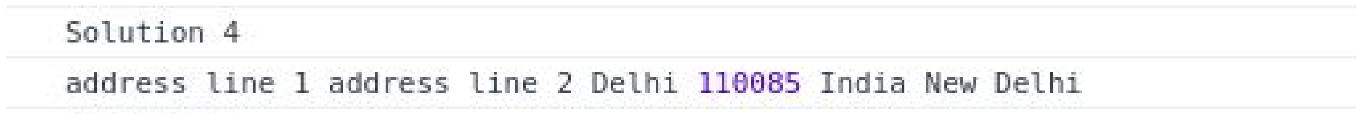
phoneNo: 9999999999

}

Solution:

```
const user = {
  firstName: 'Sahil',
  lastName: 'Dua',
  Address: {
     Line1: 'address line 1',
     Line2: 'address line 2',
     State: 'Delhi',
     Pin: 110085,
     Country: 'India',
     City: 'New Delhi',
  },
  phoneNo: 9999999999
}

const {Line1, Line2, State, Pin, Country, City} = user.Address;
console.log("Solution 4");
console.log(Line1, Line2, State, Pin, Country, City);
```

Output:



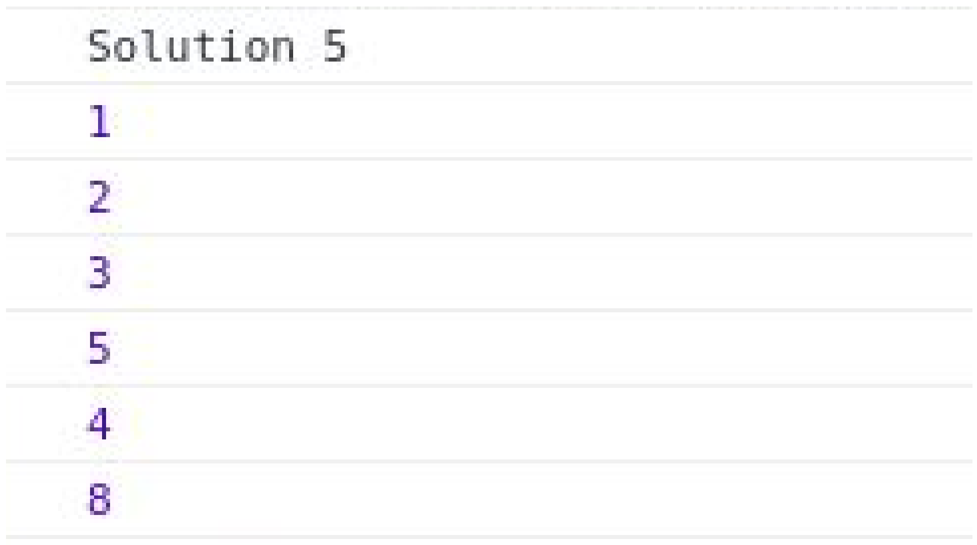
```
Solution 4
address line 1 address line 2 Delhi 110085 India New Delhi
```

Q5. Filter unique array members using Set.

Solution:

```
let sampleArray = [1,2,3,2,1,5,3,1,4,8];
let uniqueElements = new Set(sampleArray);
console.log("Solution 5");
for(let element of uniqueElements.values()){
  console.log(element);
}
```

Output:



```
Solution 5
1
2
3
5
4
8
```

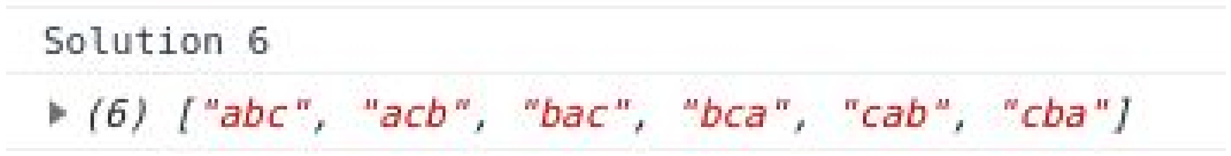
Q6. Find the possible combinations of a string and store them in a MAP?

Solution:

```
function combinations(string)
{
  var results = [];

 if (string.length === 1) {
  results.push(string);
```

```
    return results;
  }

  for (var i = 0; i < string.length; i++) {
    var char1 = string[i];
    var char2 = string.substring(0, i) + string.substring(i + 1);
    var inner = combinations(char2);
    for (var j = 0; j < inner.length; j++) {
      results.push(char1 + inner[j]);
    }
  }
  return results;
}

console.log("Solution 6");
console.log(combinations("abc"));
```

Output:



```
Solution 6
▶ (6) ["abc", "acb", "bac", "bca", "cab", "cba"]
```

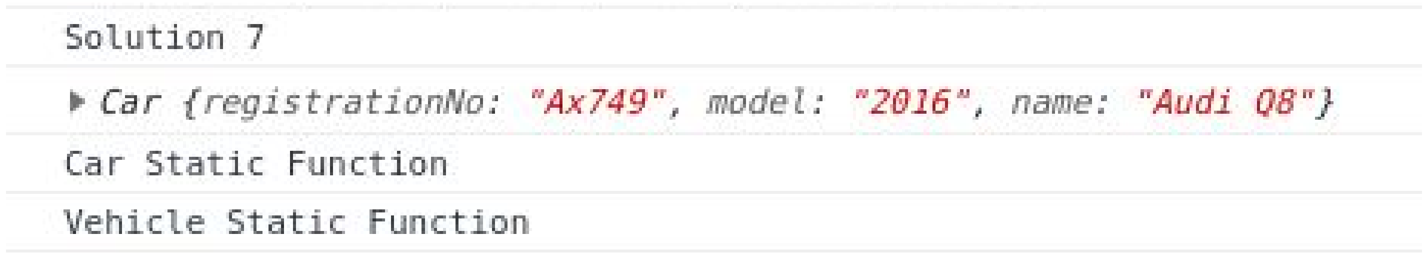
Q7. Write a program to implement inheritance upto 3 classes.The Class must public variables and static functions.

Solution:

```
class Vehicle{
   constructor(registrationNo){
       this.registrationNo = registrationNo;
   }

   static vehicleStaticFunc(){
       return "Vehicle Static Function";
   }
}

class FourWheeler extends Vehicle{
   constructor(registrationNo,model){
       super(registrationNo);
```

```
        this.model = model;
    }

    static fourWheelerStaticFunc(){
        return "Four Wheeler Static Function";
    }
}

class Car extends FourWheeler{
    constructor(registrationNo,model,name){
        super(registrationNo,model);
        this.name = name;
    }

    static carStaticFunc(){
        return "Car Static Function";
    }
}

console.log("Solution 7");
let car = new Car("Ax749","2016","Audi Q8");
console.log(car);
console.log(Car.carStaticFunc());
console.log(Vehicle.vehicleStaticFunc());
```

Output:



```
Solution 7
▶ Car {registrationNo: "Ax749", model: "2016", name: "Audi Q8"}
Car Static Function
Vehicle Static Function
```

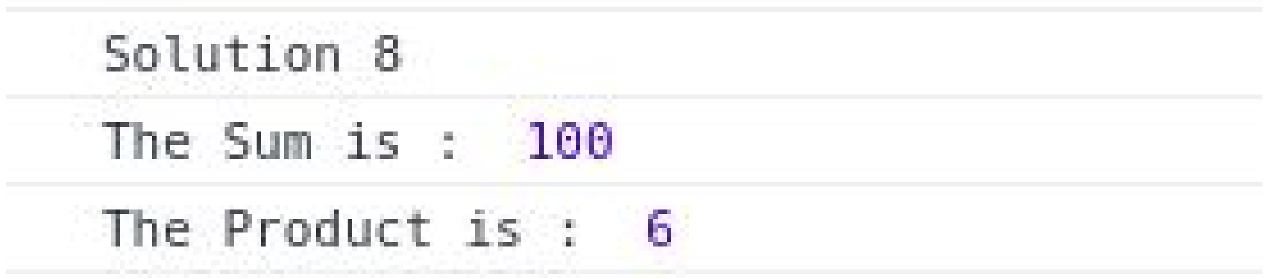
Q8. Write a program to implement a class having static functions

Solution:

```
class Calculator {
    static product(a,b){
        return a*b;
    }
```

```
  static add(a,b){
     return a+b;
  }
}
console.log("Solution 8");
console.log("The Sum is : ",Calculator.add(50,50));
console.log("The Product is : ",Calculator.product(3,2));
```

Output:



```
Solution 8
The Sum is :  100
The Product is :  6
```

Q9. Import a module containing the constants and method for calculating area of circle, rectangle, cylinder.

Solution:

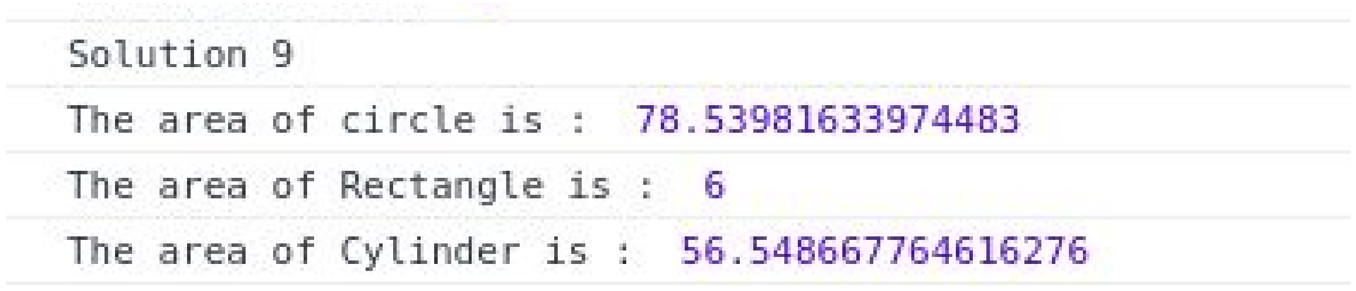
Ques9.js

```
const areaCircle = (radius)=> Math.PI*radius*radius;

const areaRectangle = (length,breadth)=>length*breadth;

const areaCylinder = (radius,height)=>Math.PI*radius*radius*height;

export{areaCircle,areaRectangle,areaCylinder};
```

index.js

```
import {areaCircle,areaRectangle,areaCylinder} from './ques9'

console.log("Solution 9");
console.log("The area of circle is : ",areaCircle(5));
console.log("The area of Rectangle is : ",areaRectangle(2,3));
console.log("The area of Cylinder is : ",areaCylinder(3,2));
```

Output:



```
Solution 9
The area of circle is :  78.53981633974483
The area of Rectangle is :  6
The area of Cylinder is :  56.548667764616276
```

Q10. Import a module for filtering unique elements in an array.

Solution:

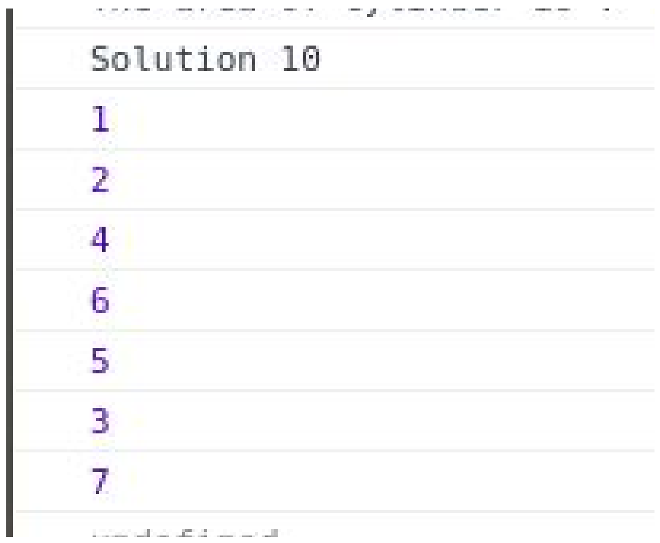
Ques10.js

```
const uniqueNumbers = (sampleArray) => {
  let uniqueElements = new Set(sampleArray);
  for(let element of uniqueElements.values()){
      console.log(element);
  }
}

export default uniqueNumbers;
```

index.js

```
import uniqueNumbers from './ques10'

let demoArray = [1,2,4,6,5,4,3,2,1,7,2,1];
console.log("Solution 10");
console.log(uniqueNumbers(demoArray));
```

Output:



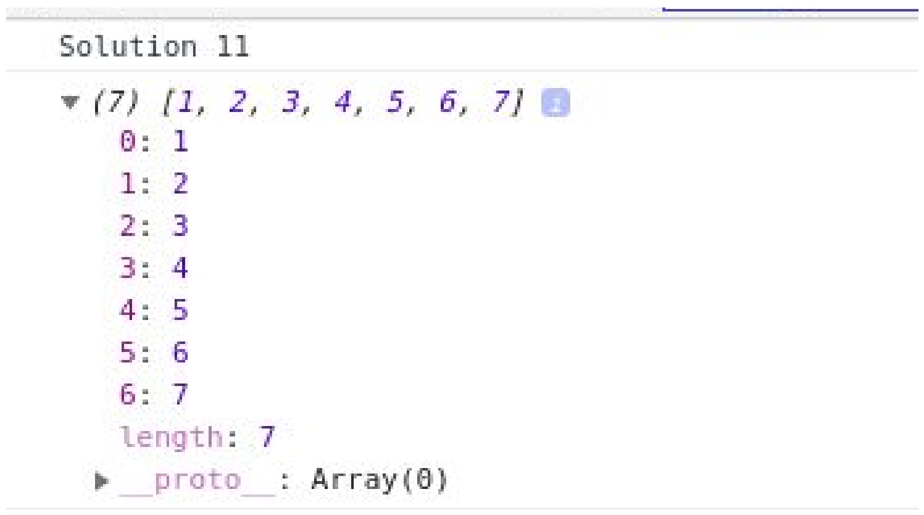
```
Solution 10
1
2
4
6
5
3
7
```

Q11. Write a program to flatten a nested array to single level.

Solution:

let sampleFlatten = [1, [2], [3, [4, [5, [6, [7]]]]]];
const flattenedArray = sampleFlatten.flat(Infinity);
console.log("Solution 11");
console.log(flattenedArray);

Output:



```
Solution 11
▼(7) [1, 2, 3, 4, 5, 6, 7]
   0: 1
   1: 2
   2: 3
   3: 4
   4: 5
   5: 6
   6: 7
   length: 7
 ▶__proto__: Array(0)
```

Q12. Implement a singly linked list in es6 and implement addFirst() addLast(), length(), getFirst(), getLast(). (without using array).

Solution:

Ques12.js

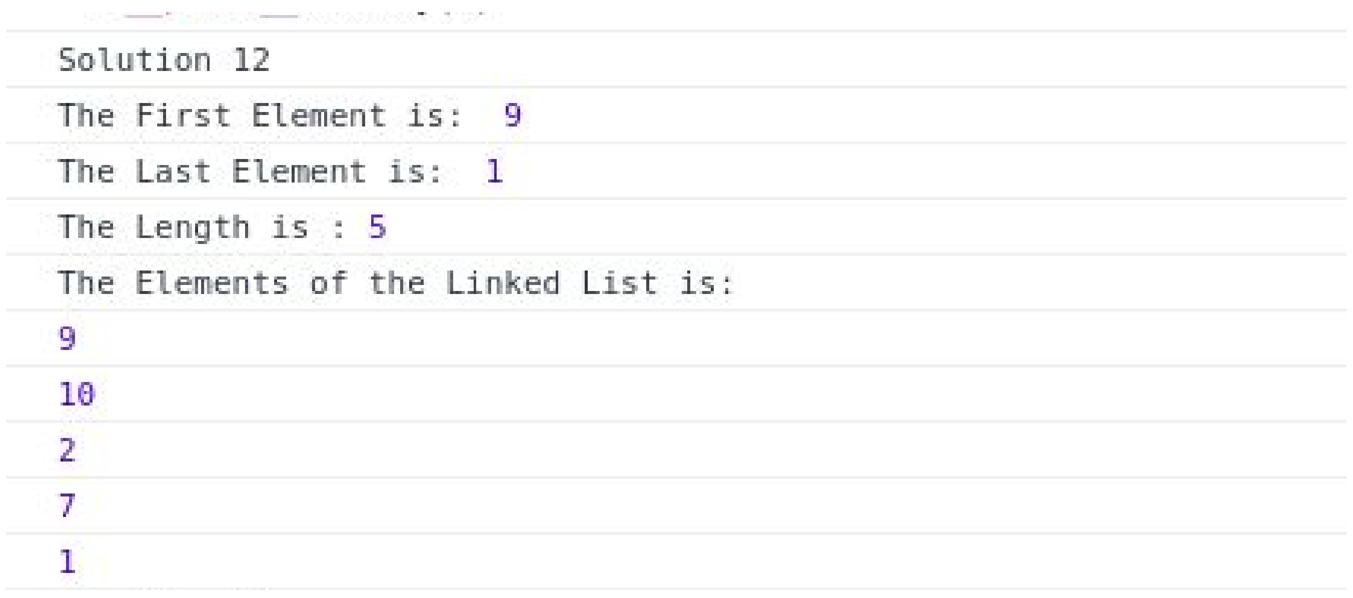
```
class Node{
  constructor(data,next=null){
     this.data=data;
     this.next=next;
  }
}

class LinkedList{
  constructor(){
      this.head = null;
  }
}

LinkedList.prototype.addFirst = function(data){
  let newNode = new Node(data);
  newNode.next=this.head;
  this.head = newNode;
  return this.head;
}

LinkedList.prototype.addLast = function(data)
{
  let newNode = new Node(data);
  if(this.head==null)
  {
     this.head=newNode;
     return this.head;
  }
  else
  {
     let temp = this.head;
     while(temp.next!=null)
     {
        temp=temp.next;

     }
     temp.next=newNode;
```

```
        return this.head;
    }
}

LinkedList.prototype.printList = function()
{
    let temp = this.head;
    while(temp.next!=null)
    {
        console.log(temp.data);
        temp=temp.next;
    }
    console.log(temp.data);
}

LinkedList.prototype.getFirst = function()
{
    let temp = this.head;
    return temp.data;
}

LinkedList.prototype.getLast = function()
{
    let temp = this.head;
    while(temp.next!=null)
    {
        temp=temp.next;
    }
    return temp.data;
}
LinkedList.prototype.getlength = function()
{
    let length =  0;
    let temp = this.head;
    while(temp.next!=null)
    {
        length = length +1;
        temp=temp.next;
    }
    console.log('The Length is :',length+1);
}

export default LinkedList;
```

index.js

```
import LinkedList from './ques12'

console.log("Solution 12");
let linkedList = new LinkedList();
linkedList.addLast(2);
linkedList.addFirst(10);
linkedList.addFirst(9);
linkedList.addLast(7);
linkedList.addLast(1);
console.log("The First Element is: ",linkedList.getFirst());
console.log("The Last Element is: ",linkedList.getLast());
linkedList.getlength();
console.log("The Elements of the Linked List is: ")
linkedList.printList();
```

Output:



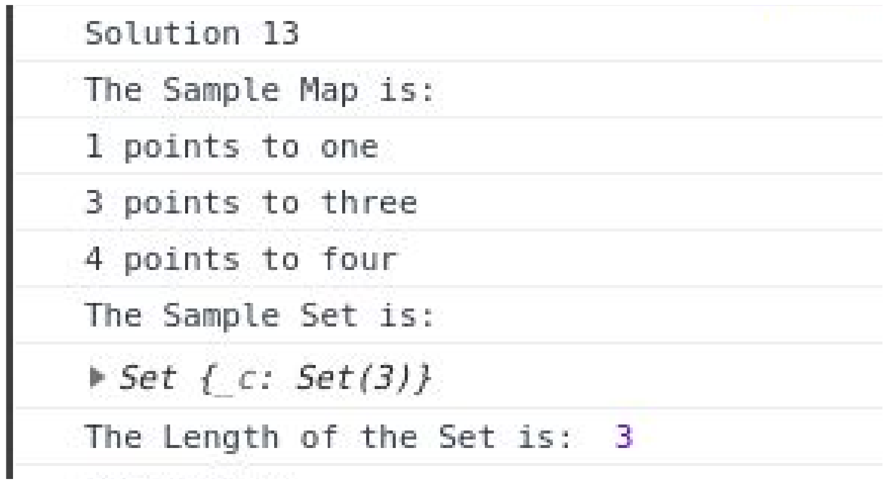
```
Solution 12
The First Element is:  9
The Last Element is:  1
The Length is : 5
The Elements of the Linked List is:
9
10
2
7
1
```

Q13. Implement Map and Set using Es6.

Solution:

```
const demoMap = function() {
  let sampleMap = new Map();
  sampleMap.set(1,"one");
  sampleMap.set(2,"two");
  sampleMap.set(3,"three");
  sampleMap.set(4,"four");
  sampleMap.delete(2);

  for(let [key,value] of sampleMap.entries()){
     console.log(`${key} points to ${value}`);
  }
}
console.log("Solution 13");
console.log("The Sample Map is: ");
demoMap();


const demoSet = function(){
  let sampleSet = new Set();
  sampleSet.add("One");
  sampleSet.add("Two");
  sampleSet.add("Three");
  sampleSet.add("Four");
  sampleSet.delete("Three");
  console.log(sampleSet);
  console.log("The Length of the Set is: ",sampleSet.size);

}
console.log("The Sample Set is: ");
demoSet();
```

Output:



```
Solution 13
The Sample Map is:
1 points to one
3 points to three
4 points to four
The Sample Set is:
▶ Set {_c: Set(3)}
The Length of the Set is:  3
```

Q14. Implementation of stack (using linked list)

Solution:

ques14.js

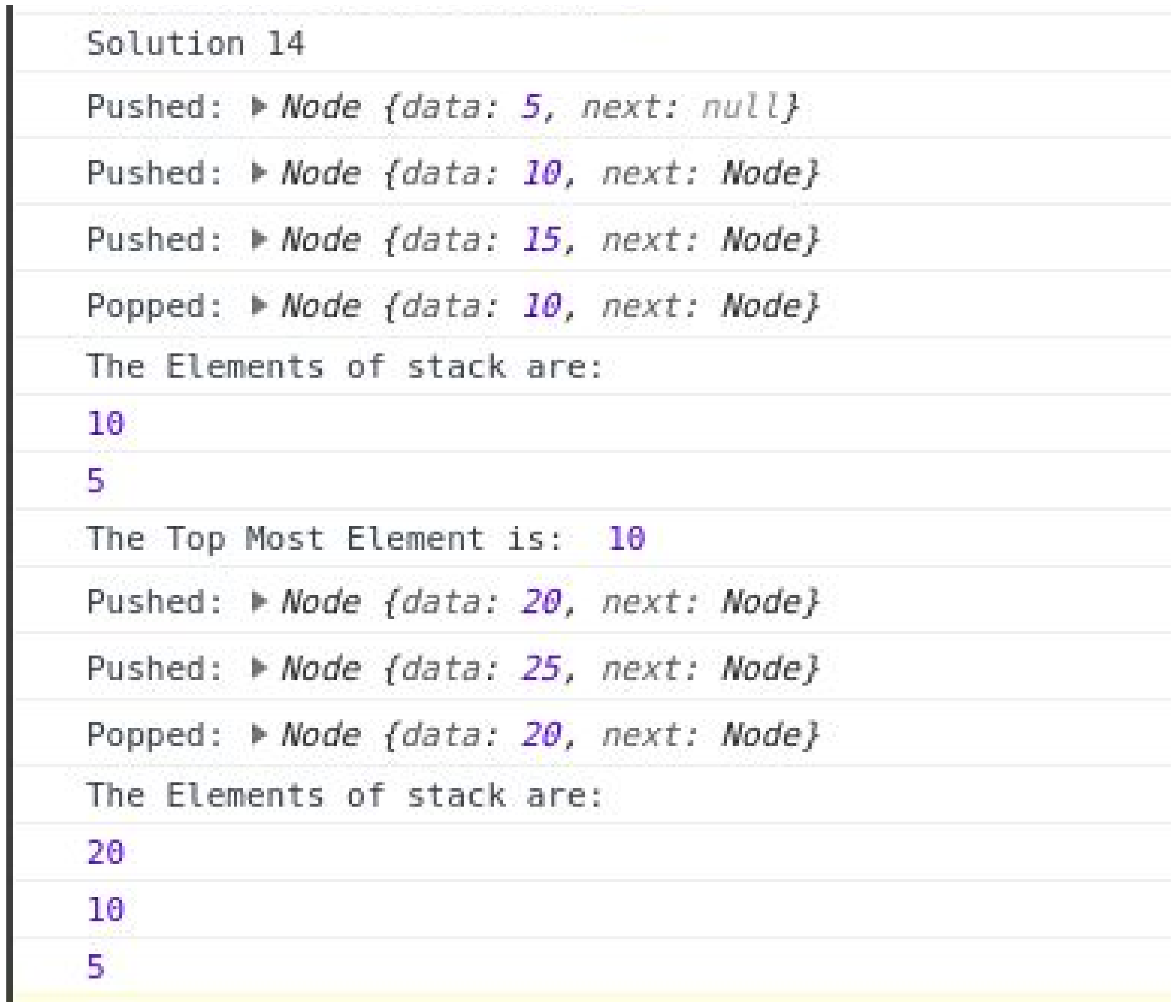
```
class Node{
  constructor(data,next=null)
  {
      this.data=data;
      this.next=next;
  }
}

class Stack{
  constructor()
  {
       this.head = null;
  }
}


let count = 0;
Stack.prototype.push = function(data){
  let newNode = new Node(data);
  newNode.next=this.head;
  this.head = newNode;
  count=count+1;
  return this.head;
}


Stack.prototype.pop = function(){
 let temp = this.head;
 temp=temp.next;
 this.head=temp;
 count=count-1;
 return this.head;
}


Stack.prototype.peek=function(){
  console.log('The Top Most Element is: ',this.head.data);
}
```

```
Stack.prototype.lengthOfStack = function(){
  console.log('The Length of stack is ' , count);
}

Stack.prototype.printStack = function()
{
   let temp = this.head;
   while(temp.next!=null)
   {
       console.log(temp.data);
       temp=temp.next;
   }
   console.log(temp.data);
}

export default Stack;
```

index.js

```
import Stack from './ques14'

console.log("Solution 14");
let stack = new Stack();
console.log('Pushed:',stack.push(5));
console.log('Pushed:',stack.push(10));
console.log('Pushed:',stack.push(15));
console.log('Popped:',stack.pop());

console.log("The Elements of stack are: ");
stack.printStack();
stack.peek();
console.log('Pushed:',stack.push(20));
console.log('Pushed:',stack.push(25));
console.log('Popped:',stack.pop());
console.log("The Elements of stack are: ");
stack.printStack();
```

Output:



```
Solution 14
Pushed: ▶ Node {data: 5, next: null}
Pushed: ▶ Node {data: 10, next: Node}
Pushed: ▶ Node {data: 15, next: Node}
Popped: ▶ Node {data: 10, next: Node}
The Elements of stack are:
10
5
The Top Most Element is:  10
Pushed: ▶ Node {data: 20, next: Node}
Pushed: ▶ Node {data: 25, next: Node}
Popped: ▶ Node {data: 20, next: Node}
The Elements of stack are:
20
10
5
```