1. Write a JS function that accepts a string as a parameter & converts the first letter of each word into upper case.

```
function capital (str){
    return str.split(" ").map(word => word.charAt(0).to
UpperCase() + word.slice(1)).join(' ');
    }
const input = 'the quick brown fox';
const result = capital (input);
console.log(result);
```

OUTPUT

The Quick Brown Fox.

2. Write a JS function that accepts a string as a parameter and finds the longest second within the string

```
function LongestW(str) {
    return str.split(' ').reduce((longest, word) =>
        word.length > longest.length ? word : longest, '');
}
const input = 'Web Development Tutorial';
const result = longestW(input);
console.log(result);
```

OUTPUT

Development

3. Write a JS function that accepts string as a parameter and counts the number of vowels within the string.

```
function countV (str){
    const vowels = ['a', 'e', 'i', 'o', 'u'];
    return str.toLowerCase().split('').reduce((count, char)
=> vowels.includes(char) ? count+1 : count, 0); }
```

```
const input = 'the quick brown fox';
const result = countV(input);
console.log(result);
```

OUTPUT

5

4 WAP JS function that returns the longest palindrome in a given string.

```
function longestPalindrome(str) {
    let longest = '';

for (let i = 0; i < str.length; i++) {
   for (let j = i; j < str.length; j++) {
      const sub = str.slice(i, j+1);
    if (isPalindrome(sub) && sub.length > longest.length) {
            longest = sub;
        }
    }
  }
  return longest;
}

function isPalindrome(str) {
      return str === str.split('').reverse().join('');
   }

const input = 'babad';
const result = longestPalindrome(input);
console.log(result);
```

OUTPUT

bab
aba

5. WA IS function that takes a callback and invokes it after a delay of 2 second.

```
function invokedelay (callback) {
        setTimeout (callback, 2000);
    }
const callback = () => console.log ('Callback invoked after 2seconds');
invokedelay (callback);
```

6. WA IS function to create a class 'Book' with properties for title, author and publication year. Include a method to display book details. Create subclass 'Ebook' that inherits from the 'Book' class. Override the display method to include the book price. Create an instance of the 'Ebook' class and display its details.

```
class Book {
    constructor (title, author, year) {
        this.title = title;
        this.author = author;
        this.year = year;
    }
    display () {
        console.log ('Title : ${this.title}, Author : ${this.author},
        Publication Year : ${this.year}');
    }
}
class Ebook extends Book {
    constructor (title, author, year, price) {
        super (title, author, year);
        this.price = price;
    }
    display () {
        super.display ();
        console.log ('Price : ${this.price}');
    }
}
```

```
const ebook = new Ebook('Haunting Adeline', 'H.D. Carlton, 2021,
                        ₹400);
ebook.display();
```

OUTPUT

```
Title: Haunting Adeline
Author: H.D. Carlton.
Publication year: 2021
Price: ₹400
```

7. WA JS program that creates a class called University with properties for university name and departments. Include methods to add department, remove department & display all departments. Create an instance of the University class & add & remove departments.

```
class University {
    constructor(name) {
        this.name = name;
        this.dept = [];
    }
    adddept(department) {
        this.dept.push(department);
    }
    Removedept(department) {
        const index = this.dept.indexOf(department);
        if (index !== -1) {
            this.dept.splice(index, 1);
        }
    }
    display() {
        console.log(`Departments of ${this.name}:`);
        this.dept.forEach((department, index) => {
            console.log(`${index+1}. ${department}`);
        });
    }
}
```

```
const ...                                    ('Harvard University');
const university = new University
university.addDept('Computer Science');
university.addDept('Mathematics')
university.addDept('Physics');
university.addDept('Biology');
university.display();
university.removeDept('Physics');
university.display();
```

8 WA JS program that creates a class called BankAccount with properties for account number, account holder name and balance. Include method to deposit, withdraw & transfer money between accounts. Create multiple instances of the BankAccount class & perform operations such as depositing, withdrawing & transferring money.

```
class BankAccount {
    constructor (num, holder, balance) {
        this.accNum = num;
        this.accHold = holder;
        this.balance = balance;
    }
deposit (amount) {
    if (amount > 0) {
        this.balance += amount;
        console.log ('Deposited ${amount} into ${this.
        accHold}'s account.');
    }
    else {
        console.log ('Deposit should be more than zero');
    }
}

withdraw (amount) {
    if (amount > 0 && amount <= this.balance) {
        this.balance -= amount;
        consol.log ('Withdrawn ${amount} from ${this.
        accHold}'s account.');
    }
```

```
else {
    console.log('Invalid');
    }
}
transfer(amount, recipient) {
    if (amount > 0 && amount <= this.balance) {
        this.balance -= amount;
        recipient.deposit(amount);
        console.log(`Transferred ${amount} to ${
        recipient.accHold}'s account.`);
    }
else {
    console.log('Invalid');
    }
}
display() {
    console.log(`${this.accHold}'s account balance : ${this.balance}`
    );
    }
}
const acc1 = new BankAccount(123456, 'JK', 1000);
const acc2 = new BankAccount(987654, 'THV', 500);

acc1.display();
acc2.display();
acc1.deposit(500);
acc1.display();
acc1.withdraw(200);
acc1.display();
```

9 WAP in Javascript to create a class 'Vehicle' with properties for make, model & year. Include a method to display vehicle details. Create a subclass called 'Car' that inherits from the 'Vehicle' class & include an additional property for the number of doors. Override the display method to also include the number of doors.

t ebook = new Ebook('Haunting Adeline' . H.R ...

```
class Vehicle
  constructor (make, model, year) {
        this. make = make;
        this. model = model;
        this. year = year;
     }

  display () {
      console log ('Vehicle: ${this. year} ${this. make}
              ${this. model}');
        }
   }

class Car extend Vehicle {
      constructor (make, model, year, doors) {
        super (make, model, year);
        this. doors = doors;
      }
   display() {
    console. log ('Car : ${this. year} ${this. make} ${this. model}
        ${this. doors} doors');
      }
    }
const vehicle1 = new Vehicle ('Toyota', 'Camry', 2020);
vehicle1. display ();
const car1 = new Car ('Honda', 'Civic', 2018, 4);
car1. display ();
```

12. WA JS program that creates a class called 'Shape' with a method to calculate the area. Create two subclasses. 'Circle' & 'Triangle', that inherit from the 'Shape' class & override ~~the~~ the area calculation method. Create an instance of the 'Circle' class & calculate its area. Similarly, do the same for 'Triangle' ~~to~~ class.

```
else { console.log ('Invalid');

class Shape {
    Area () {
        return 0;
    }
}

class Circle extends Shape {
    constructor (radius) {
        super ();
        this.radius = radius;
    }

    Area () {
        return Math.PI * this.radius * this.radius;
    }
}

class Triangle extends Shape {
    constructor (base, height) {
        super ();
        this.base = base;
        this.height = height;
    }

    Area () {
        return (this.base * this.height) / 2;
    }
}

const circle = new Circle (5);
const AreaC = circle.Area ();
console.log (`Area : ${AreaC.toFixed (2)}`);

const triangle = new Triangle (4, 6);
const AreaT = triangle.AreaC ();
console.log (`Area : ${AreaT}`);
```