**EXERCISE2:**

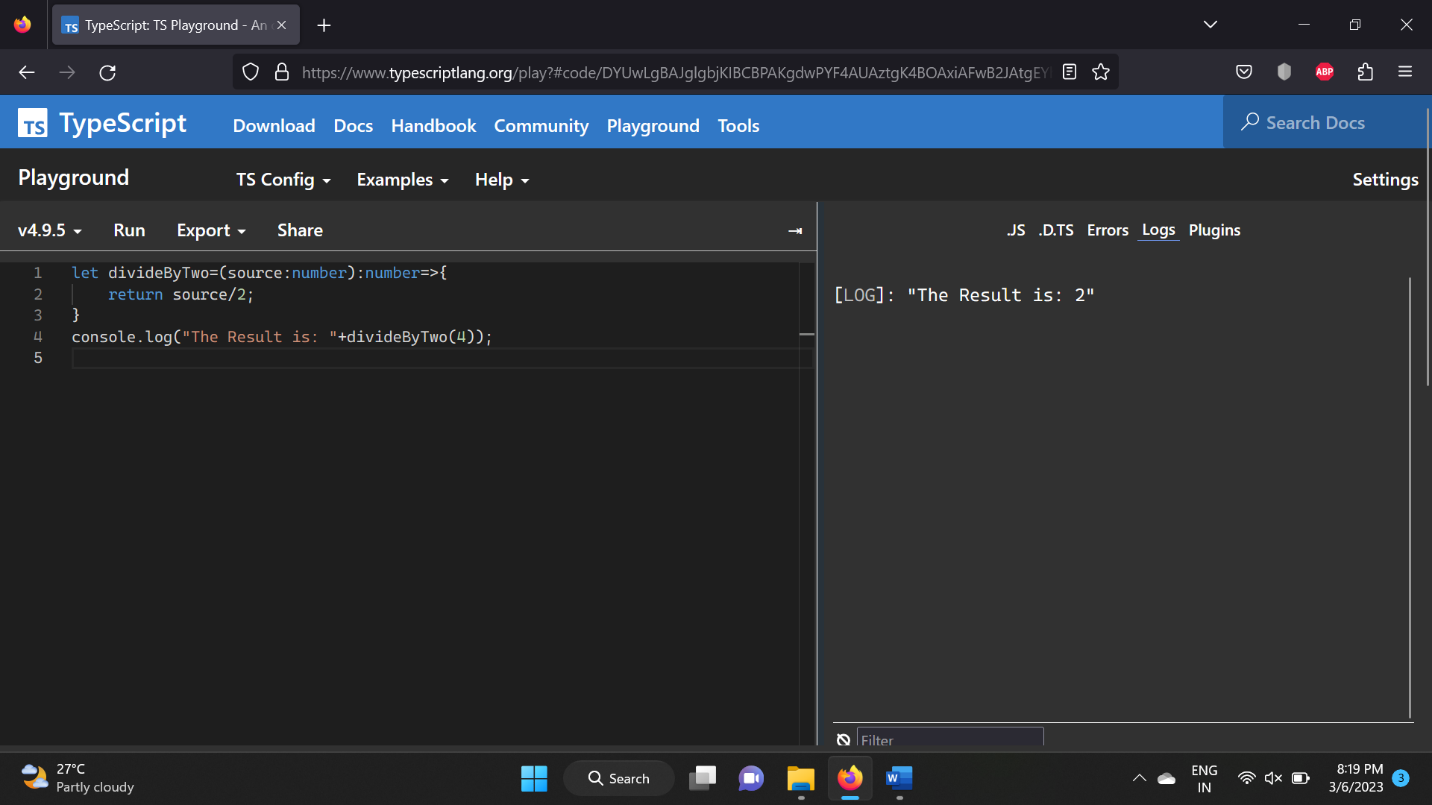
**Question1:**

let divideByTwo=(source:number):number=>{

    return source/2;

}

console.log("The Result is: "+divideByTwo(4));



**Question2:**

let checkVowel=(data:string):boolean =>{

    let vow=["a","e","i","o","u","A","E","I","O","U"];

    for(var val in vow){

        if(vow[val]==data)

        return true;

    }

    return false;

}

function verify(data:string){

    if(checkVowel(data)==true)

    console.log("It is a vowel");

    else

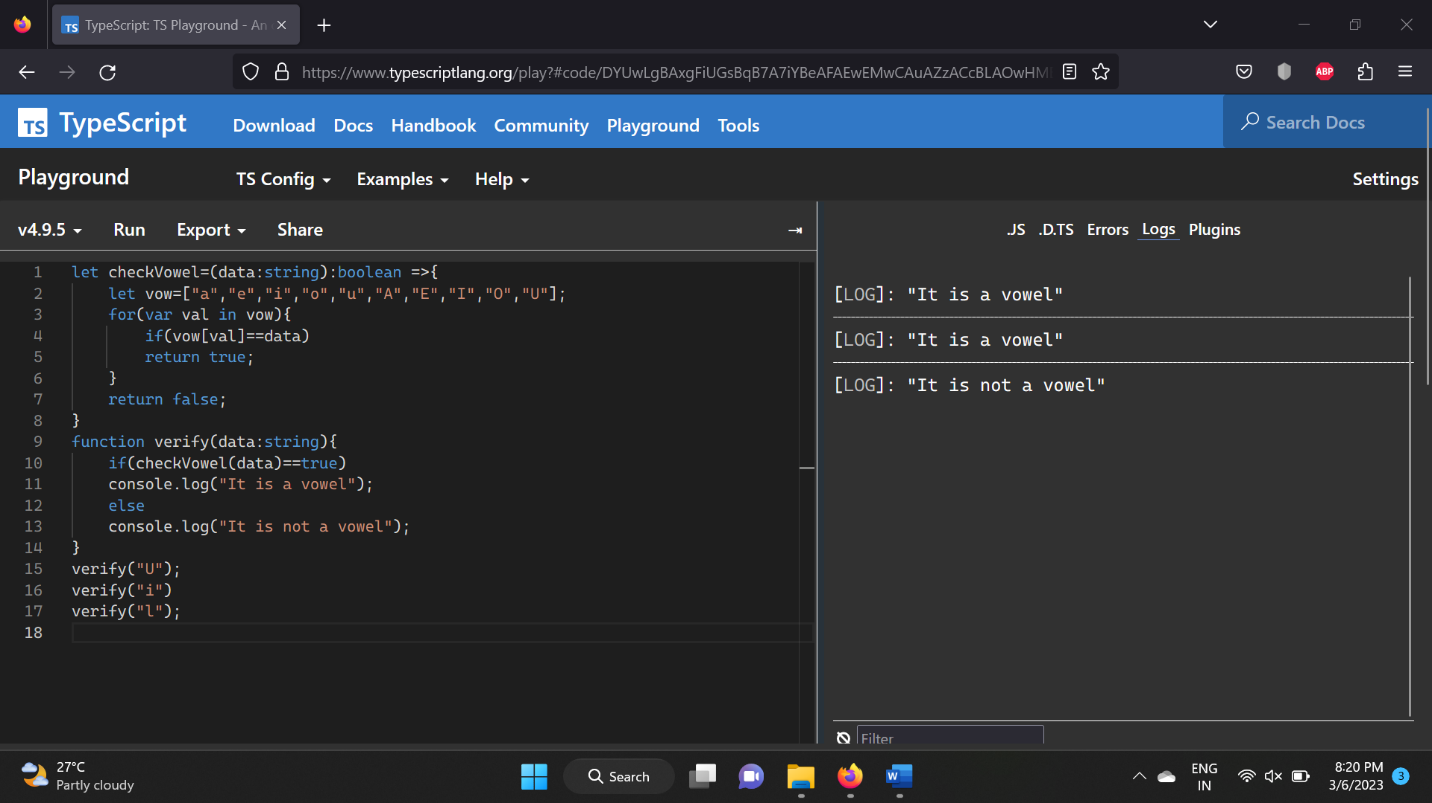
    console.log("It is not a vowel");

}

verify("U");

verify("i")

verify("l");



**Question3:**

function prodBill(custName:string,...products:number[]){

console.log("The Customer Name is: "+custName);

let val:number=0;

for(var i in products){

    val+=products[i];

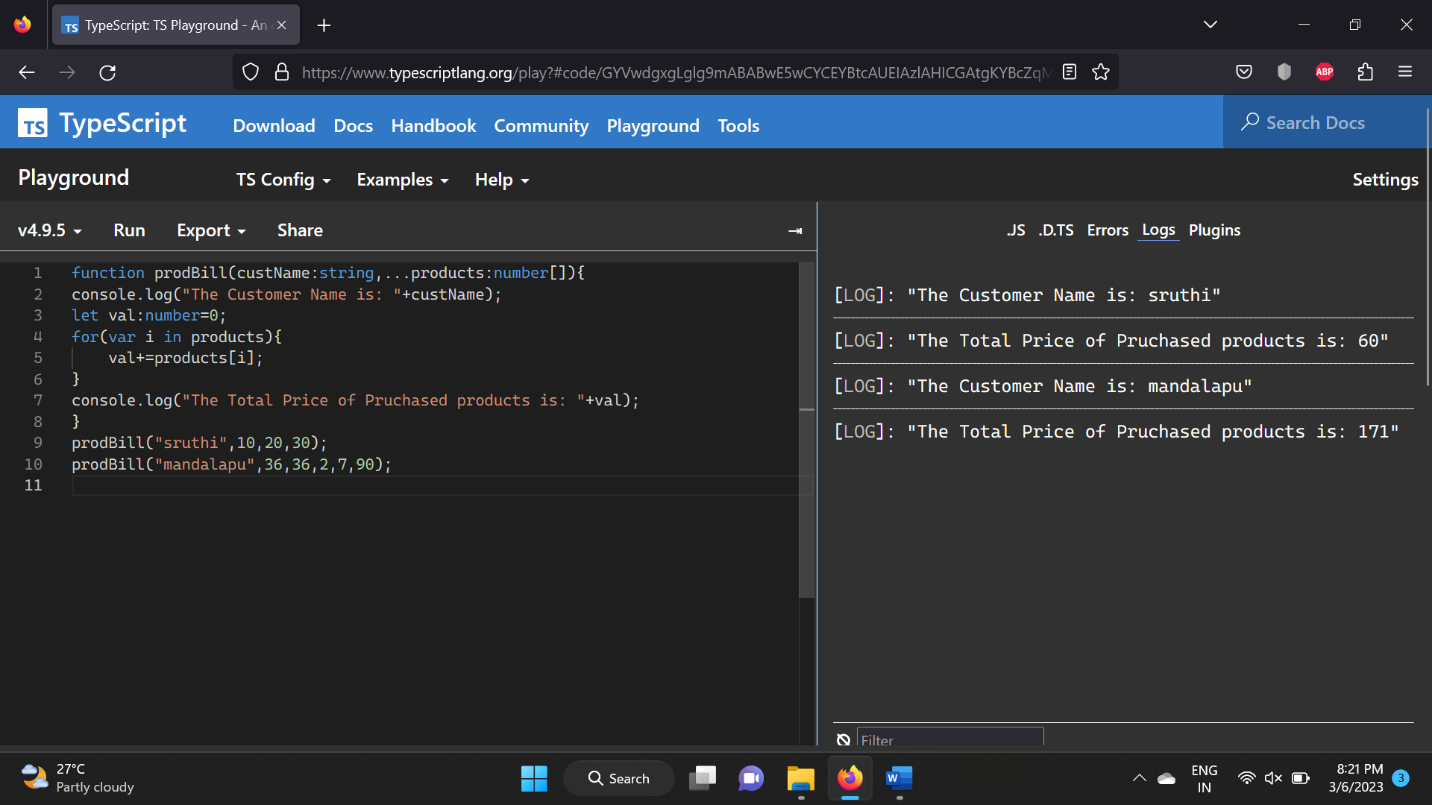
}

console.log("The Total Price of Pruchased products is: "+val);

}

prodBill("sruthi",10,20,30);

prodBill("mandalapu",36,36,2,7,90);



**Question4:**

function stdInfo(stdName:string,...scores:number[]){

    console.log("The name of the student is: "+stdName);

    let val:number=0;

    for(var i in scores){

        val+=scores[i];

    }

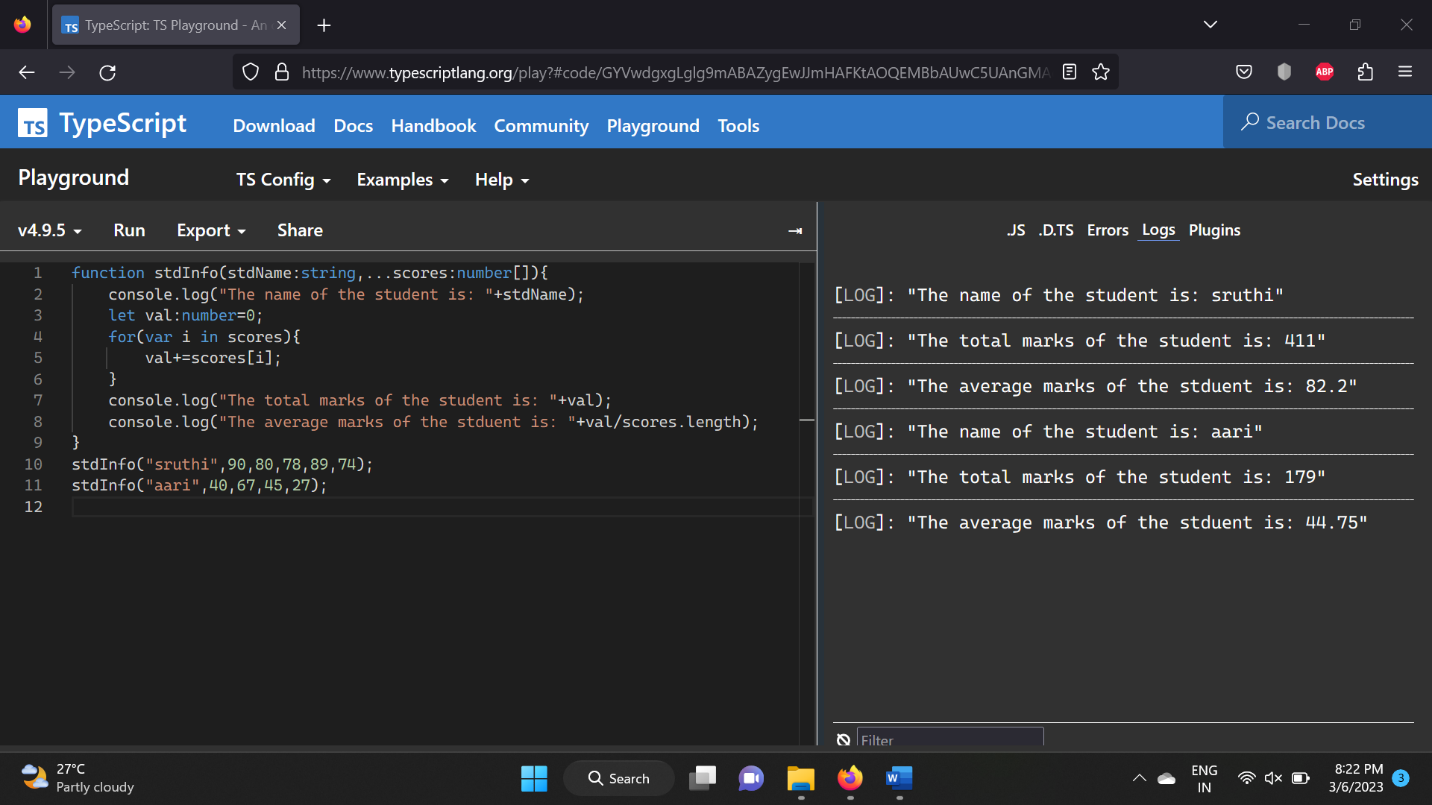
    console.log("The total marks of the student is: "+val);

    console.log("The average marks of the stduent is: "+val/scores.length);

}

stdInfo("sruthi",90,80,78,89,74);

stdInfo("aari",40,67,45,27);



**Question5:**

interface IceCream{

    icecreamFlavour:string;

    scoops:number;

}

**Question6:**

interface IceCream{

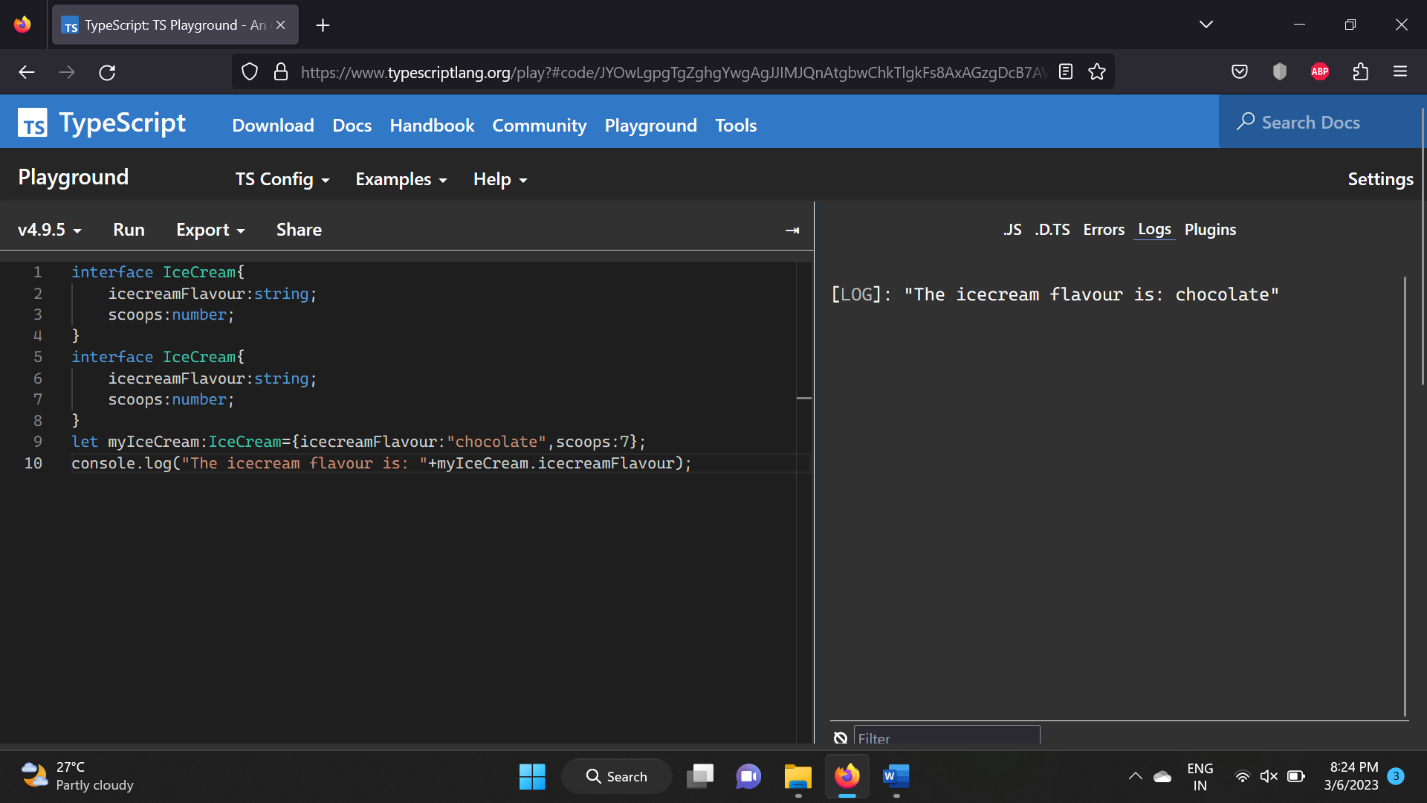
    icecreamFlavour:string;

    scoops:number;

}

let myIceCream:IceCream={icecreamFlavour:"chocolate",scoops:7};

console.log("The icecream flavour is: "+myIceCream.icecreamFlavour);



**Question7:**

interface IceCream{

    icecreamFlavour:string;

    scoops:number;

}

let myIceCream:IceCream={icecreamFlavour:"chocolate",scoops:2};

console.log("The icecream flavour is: "+myIceCream.icecreamFlavour);

let dessert:IceCream={icecreamFlavour:"strawberry",scoops:7};

function tooManyScoops(dessert:IceCream){

    if(dessert.scoops>=4)

    console.log("Too many scoops!");

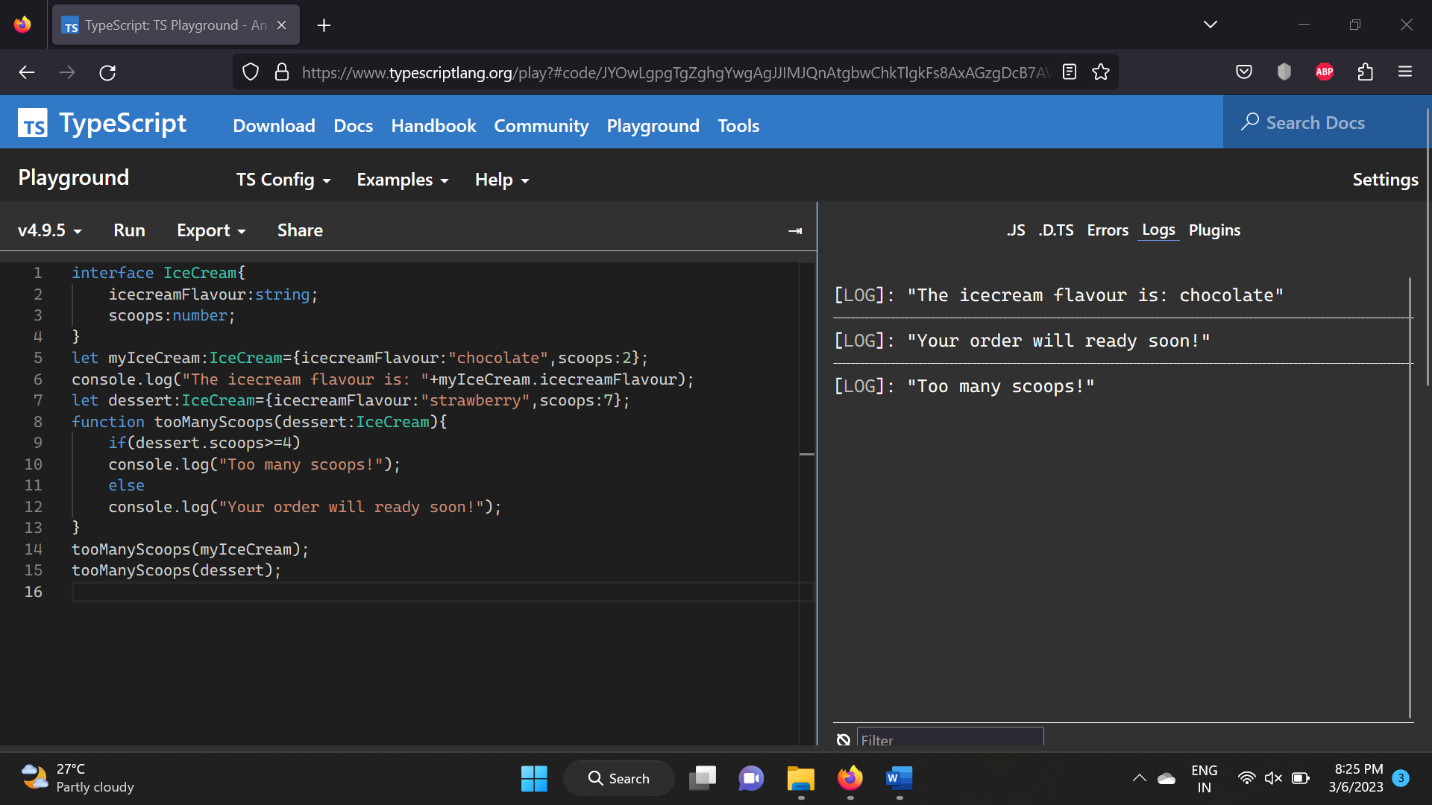
    else

    console.log("Your order will ready soon!");

}

tooManyScoops(myIceCream);

tooManyScoops(dessert);



**Question8:**

interface IceCream{

    icecreamFlavour:string;

    scoops:number;

}

interface Sundae extends IceCream{

    sauce:string;

    nuts?:boolean;

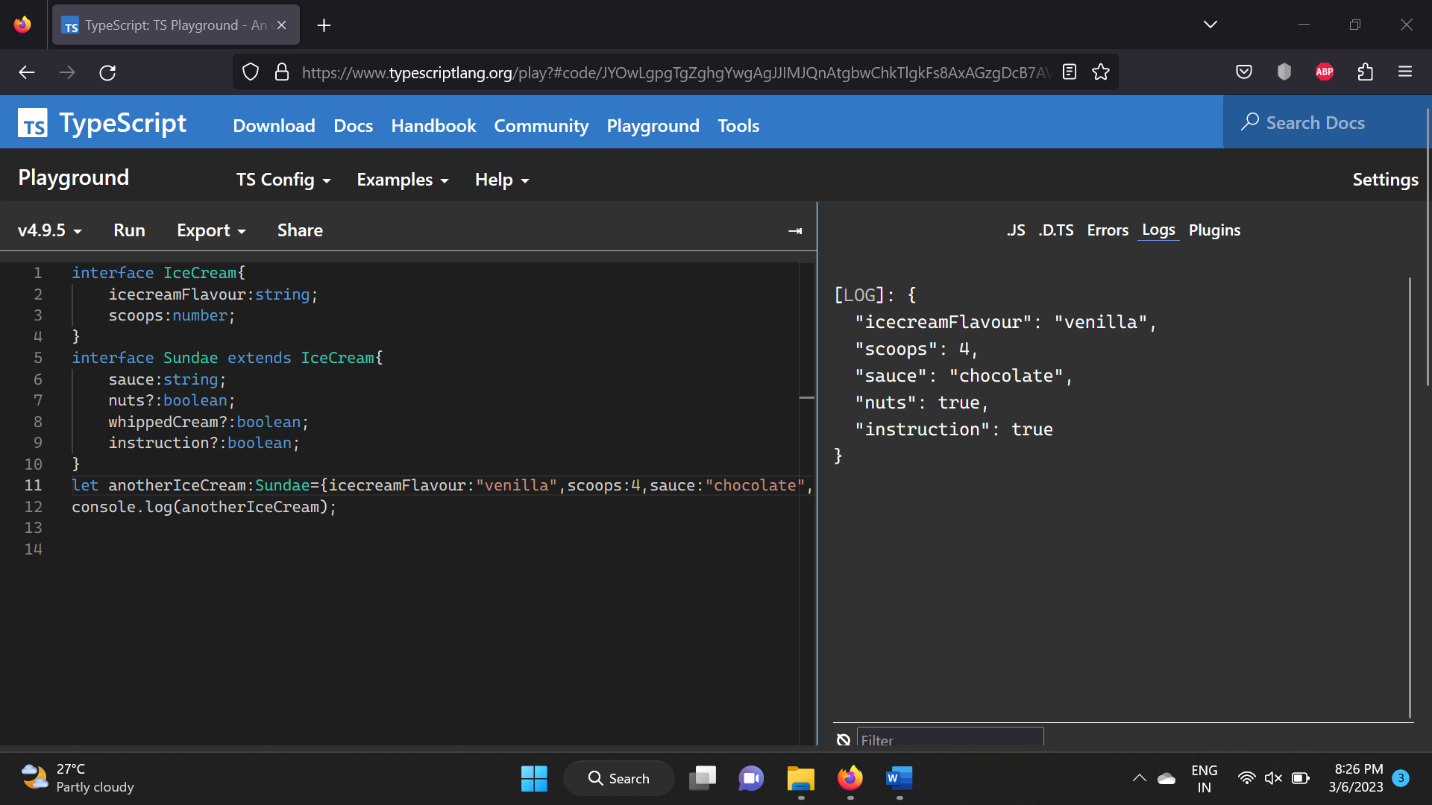
    whippedCream?:boolean;

    instruction?:boolean;

}

let anotherIceCream:Sundae={icecreamFlavour:"venilla",scoops:4,sauce:"chocolate",nuts:true,instruction:true};

console.log(anotherIceCream);



**Question9:**

class Car{

    \_make:string;

    \_color:string;

    \_doors?:number;

    constructor(\_make:string,\_color:string,\_doors?:number){

        this.\_make=\_make;

        this.\_color=\_color;

        if(\_doors==undefined)

        this.\_doors=4;

        else

        this.\_doors=\_doors;

    }

    accelerate(speed:number):string{

        return "The speed of the car is: "+speed;

    }

    brake():string{

        return "function brake is invoked";

    }

    turn(direction:'left'|'right'){

        return "The direction to be turned is: "+direction;

    }

    worker():string{

        return "function worker is invoked";

    }

}

let c1:Car=new Car("doors","red",3);

console.log(c1);

console.log(c1.accelerate(60));

console.log(c1.brake());

console.log(c1.turn("left"));

console.log(c1.worker());

let c2:Car=new Car("doors","blue");

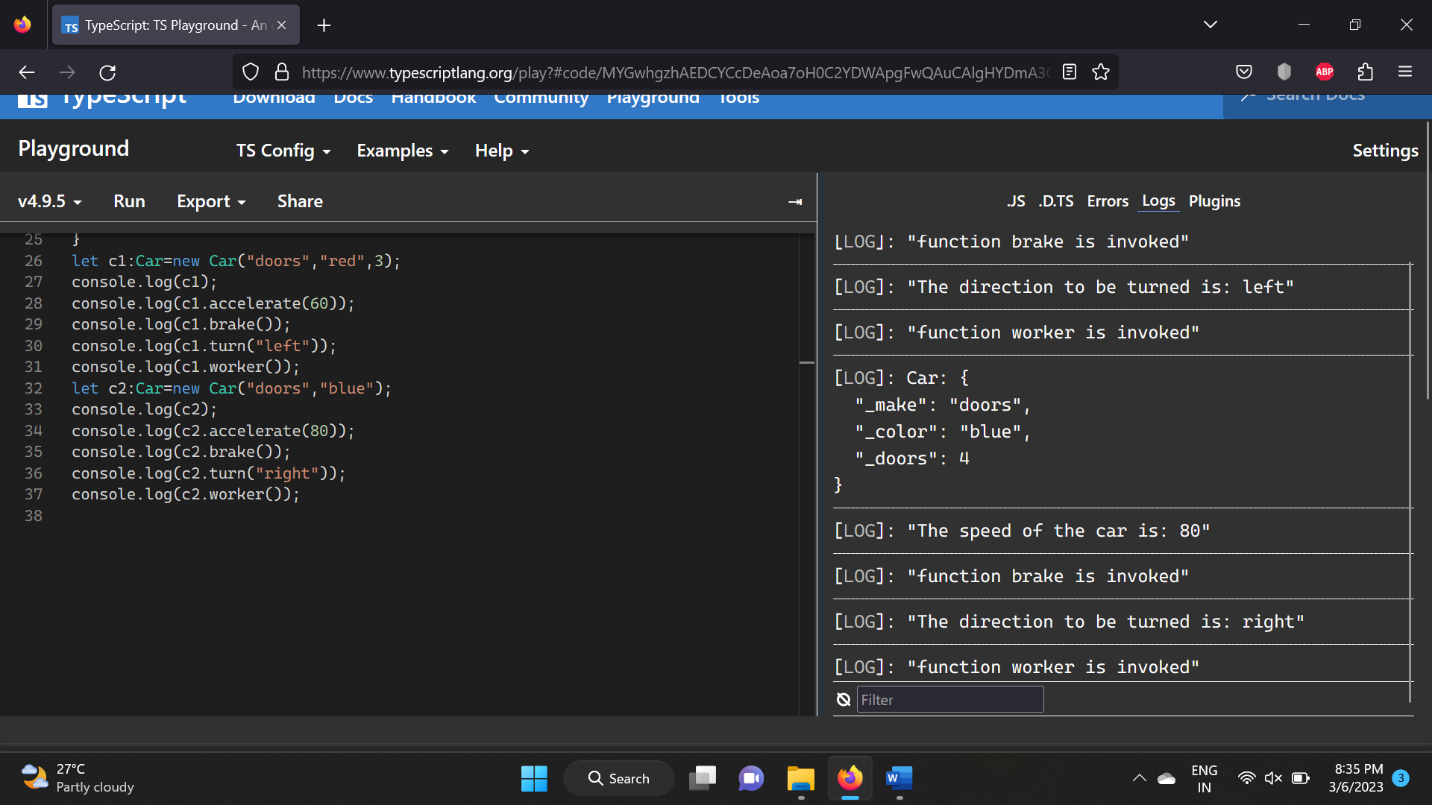
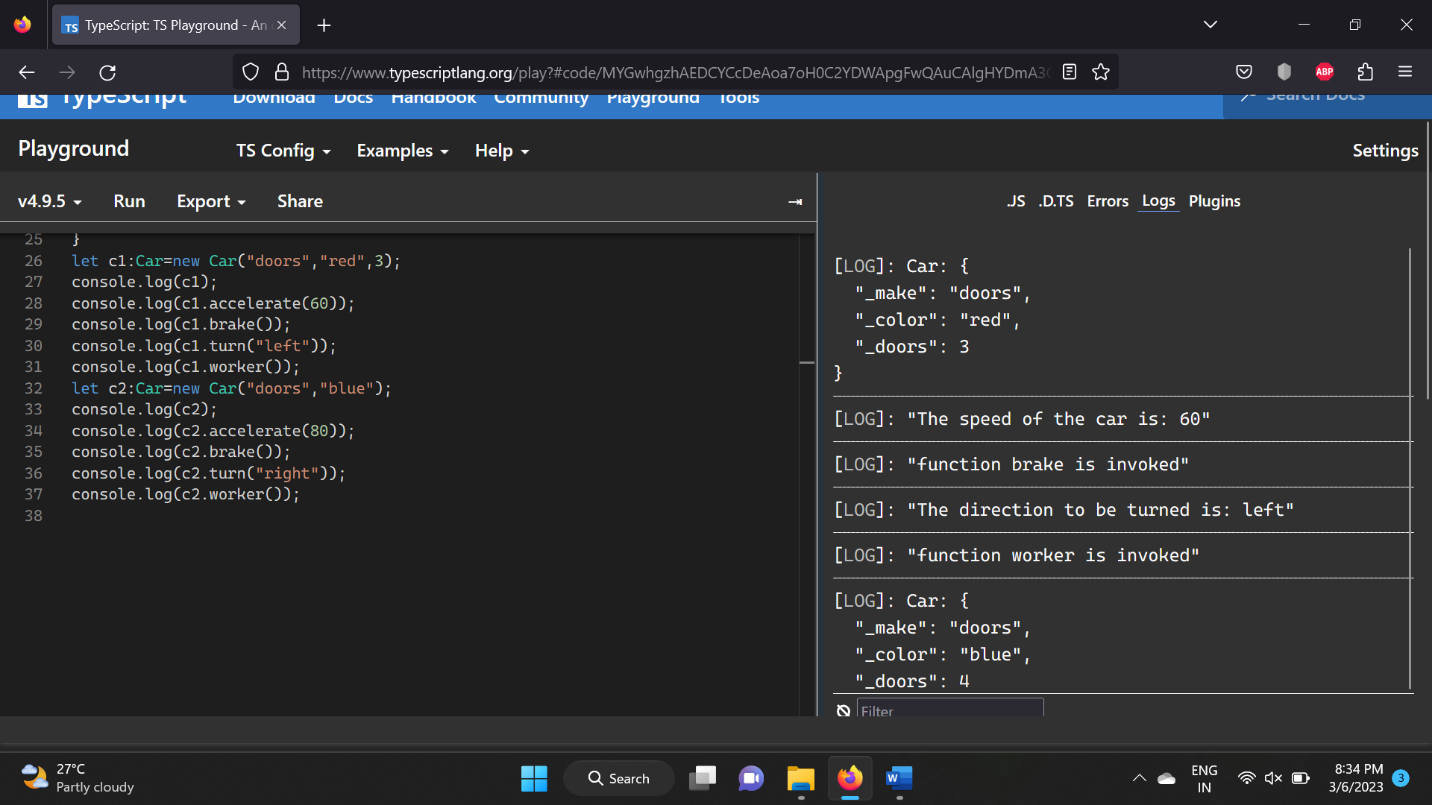
console.log(c2);

console.log(c2.accelerate(80));

console.log(c2.brake());

console.log(c2.turn("right"));

console.log(c2.worker());



**Question10:**

class Car{

    \_make:string;

    \_color:string;

    \_doors?:number=4;

    constructor(\_make:string,\_color:string,\_doors?:number){

        this.\_make=\_make;

        this.\_color=\_color;

        this.\_doors=\_doors;

    }

}

let myCar1:Car=new Car("doors","yash");

console.log("The color of car is: "+myCar1.\_color);

