**Assignment – 5**

Problem Statement: Interface, Class

1. Create a product type object interface with id, name, description, and price fields, create an object, Define optional properties, Create a compile time.
2. interface Product{
3. id:number,
4. name:string,
5. description:string,
6. price?:number
7. }
8. var prod:Product={
9. id:12345,
10. name:"Vivo",
11. description:"Good",
12. price:10000
13. }

2. Perform Addition, subtraction function by functional interfaces in typescript, define a return type to the functional interface.

interface doAirthmetic{

    add(x:number,y:number):number;

    sub(x:number,y:number):number;

}

class perform implements doAirthmetic{

    x:number;

    y:number;

    add(n1:number,n2:number):number{

        return n1+n2;

    }

    sub(n1:number,n2:number):number{

        return n1-n2;

    }

}

var obj=new perform();

console.log("Addition: ",obj.add(12,23));

console.log("Subtraction: ",obj.sub(10,5));

Graphical user interface, text, application, email

Description automatically generated

3. Create a numberIndex.js file and Perform sample Array interface using number index and string index.

**//String indexed Array Interface**

interface FlowerProperty{

    [flower:string]:string;

}

var obj1:FlowerProperty={};

obj1["Rose"]="Red";

obj1["Jasmine"]="White";

obj1["Lotus"]="Pink";

for(var item in obj1){

    console.log(item,"Color: ");

    console.log(obj1[item]);

}

**//Number indexed Array Interface**

interface FlowerProperty{

    [cost:number]:string;

}

var obj1:FlowerProperty=["Rose","Jasmine"];

for(var item in obj1){

    console.log(item);

    console.log(obj1[item]);

}

Number Indexed String Indexed­­

Graphical user interface, text, application, email

Description automatically generated Graphical user interface, text, application, email

Description automatically generated

4. Create a passenger class and Implement inheritance by creating a base class and child class with properties and methods using typescript.

class Passenger{

    name:string;    seatNo:number;

    from:String;    to:string;

    constructor(name:string,seatNo:number,from:String,to:string){

        this.name=name;this.seatNo=seatNo;

        this.from=from;this.to=to;

    }

}

class details extends Passenger{

    constructor(name:string,seatNo:number,from:string,to:string){

        super(name,seatNo,from,to);

    }

    display(){

        console.log("Name: ",this.name,"\nSeatNo: ",this.seatNo,"\nFrom: ",this.from,"\nto: ",this.to);

    }

}

var p=new details("John",1,"Vijayawada","Hyderabad");

p.display();

Graphical user interface, text, application

Description automatically generated

5. Create a flight interface with 3 properties of ‘flight No, from, and to’, implement the interface.

interface Flight{

    flightNo:number;     from:String;    to:string;

    display();

}

class child implements Flight{

    flightNo:number;

    from:String;

    to:string;

    constructor(flightNo:number,from:String,to:string){

        this.flightNo=flightNo;

        this.from=from;this.to=to;

    }

    display(){

        console.log("FlightNo: ",this.flightNo,"\nFrom: ",this.from,"\nto: ",this.to);

    }

}

var p=new child(1,"Banglore","Hyderabad");

p.display();

Graphical user interface, text, application

Description automatically generated