<Q> what is the output of the below given code snippet?

```
let name: string = `  HSBC     `;

let year: number = 5;

let sentence: string = `Hello, I work in ${ name }.

I'll be completing ${ year + 1} years next month.`;

console.log(sentence)
```

a) Hello, I work in HSBC .
   I'll be completing 6 years next month.
b) Hello, I work in name.
   I'll be completing year + 1 years next month.
c) Hello, I work in ${ name }.
   I'll be completing ${ year + 1} years next month.
d) Hello, I work in undefined.
   I'll be completing NAN years next month.

<Q>Which is/are the correct Expression/s to define the following function as Arrow function?

```
var calculateInterest = function (amount, interestRate, duration) {

return amount * interestRate * duration / 12;

}
```

1. var calculateInterest = (amount, interestRate, duration) => amount * interestRate * duration / 12;

2. var calculateInterest = (amount, interestRate, duration) => {

return amount * interestRate * duration / 12;}

3. var calculateInterest = function (amount, interestRat, duration) => amount * interestRate * duration / 12;

a) Both 1 and 2.
b) All of these.
c) Both 1 and 3.
d) Only 2.

<Q>What is the output of below code:

```
1.  var addition = (x:number)=> x=x+10;
2.
3.  var display = (x)=>{
4.                  if(typeof x=="number"){
5.                  addition(x);
6.                  console.log(x);
7.                  console.log(x+" is numeric")
8.          } else if(typeof x=="string"){
9.                  console.log(x+" is a string")
10.         }
11.}
12.
13.
14. //function Call
15.
16. display("Jason");
17. display(123);
```

a) Jason is a string
   123 is numeric
   133

b) Jason is a string
   133
   123 is numeric

c) 133 is numeric
   133
   Jason is a string

d) Jason is a string
   123
   123 is numeric

e) Not Answered

<Q> Predict the output of the below code.

```
interface Product{
        productId?:number;
        productName:string;
}

function
```

```
getProductDetails(productobj:Product={productName:'Mobile',productCategory:'Gadget'})
:string
{
        return "The product name is "+productobj.productName;

}
let productDetails:string=getProductDetails();
console.log(productDetails);
```

a) The product name is undefined
b) The product name is Mobile
c) Compilation Error because incorrect value is assigned to default parameter
d) None of these

<Q> Observe the code snippet and choose the right answer from the options provided

```
class University{
public universityCode : number;
protected courseCode: number;
protected status: string;
constructor(universityCode:number, courseCode:number){
        this.universityCode = universityCode;
        this.courseCode = courseCode
}
}
class College extends University{
private collegeId: number;
    constructor(universityCode:number, courseCode:number, collegeId:
number) {
        super(universityCode, courseCode);
        this.collegeId = collegeId;
    }
    getCollegeDetails():void{
        console.log(this.collegeId+" offers the course
"+this.courseCode);
    }
}
var college: College = new College(2001, 403, 23);
console.log("University Code :"+college.universityCode)
college.getCollegeDetails();
```

a) Compilation Error
b) University Code :undefined
23 offers the course 403
c) University Code :2001
23 offers the course 403
d) University Code :undefined
23 offers the course undefined

<Q> What is the output if the below code is executed

```
class Lathe{
        static latheCount : number = 100;
        static updateLatheCount():number{
                return(Lathe.latheCount++);
}
getLatheName():string{
        return("Lathe Name : MYSLATHE"+Lathe.updateLatheCount());
}
}
var lathe:Lathe = new Lathe();
console.log(lathe.getLatheName());
```

a) Error: static method cannot be accessed by non static method
b) Error: static variable once initialied cannot be changed
c) LatheName : MYSLATHE101
d) LatheName : MYSLATHE100

<Q> Consider the below code

```
class Book {
   title: string;
   price: number;
   author: string;
```

```
    constructor(title, price, author, public publisher?) {
      this.title = title;
      this.price = price;
      this.author = author;
    }

    display() {
      console.log(this.title, this.price, this.author, this.publisher)
    }

}

let book1 = new Book("Typescript Basics", 100, "ETA", "  HSBC     ");
book1.display();
let book2 = new Book("Angular Basics", 200, "ETA");
book2.display();
```

What will be the console output of the above code

a) error: this.publisher is not a instance varianle in display method
b) error: constructor cannot have a optional parameter
c) error: public parameter cannot be optional
d) Typescript Basics 100 ETA HSBC
Angular Basics 200 ETA undefined

<Q> Predict the output of the below code.

```
1.  interface Player
2.  {
3.    playerId:string;
4.    play();
5.  }
6.  class Team
7.  {
8.        teamId:string;
9.        playerId:string;  //captain id
10.       play()
11.       {
```

```
12.                         console.log('I am playing in a Team')
13.                 }
14.         }
15.
16.         class GameClass extends Team implements Player
17.         {
18.
19.         }
20.
21.         new GameClass().play();
22.
```

a) Compilation Error at line-16 ‘play() method of Player is not implemented in GameClass’
b) Compilation Error at line-16 ‘Property playerId is missing in type GameClass’
c) Compilation Error at line-21 ‘play() method can not be found in type GameClass’
d) I am playing in a Team

<Q>

Predict the output of the following code snippet used to validate the employee ID and employee Mail Id of the employees of Infosoft.

```
interface Employee{
        employeeId: number;
        employeeName: string;
        employeeMailId: string;
}
function validateEmployee(employee:Employee):string{
let empIdValidation : boolean = employee.employeeId > 100000 &&
 employee.employeeId <= 999999;
let emailIdValidation : boolean = employee.employeeMailId.split('@')[1]=="infosoft.com";
         if (empIdValidation && emailIdValidation)
                 var result: string = "Details of " + employee.employeeName + " validated
succesfully";
          else
                 var result: string = "Validation of " + employee.employeeName + " unsuccesfull";
        return result;
}

let employee = {employeeId:343233, employeeName:"Arun Kumar", employeeMailId :
"arunk@infosoft.com", employeePhone : ”9767543357”};
console.log(validateEmployee(employee));
```

a) Details of Arun Kumar validated succesfully
b) Error: variable result is declared inside if/else block cannot be returned outside the block
c) Validation of Arun Kumar unsuccesfull
d) Error: employee object has more parameters than the interface

<Q> Predict the ouput

```
interface Customer{
        customerId : string;
        customerName: string;
}
interface RegularCustomer{
        retailCardNo : number;
        retailOutletId : string;
}
interface PrivilegeCustomer extends Customer, RegularCustomer{
        festiveBonus: number;
}
function fetchCreditLimit(privilegeCustomer:PrivilegeCustomer):number{
        if(privilegeCustomer.retailCardNo >= 1000 && privilegeCustomer.retailCardNo <= 5000)
return privilegeCustomer.festiveBonus*3;
        else
return privilegeCustomer.festiveBonus*2;

}
let customerOne : PrivilegeCustomer={
        customerId : "C13132",
        customerName : "Umesha",
        retailCardNo : 1230,
        retailOutletId : "R1001",
        festiveBonus : 2500
}
console.log("Festive Credit Limit :Rs."+fetchCreditLimit(customerOne)+"/-");
```

a) Festive Credit Limit :Rs.7500/-
b) Festive Credit Limit :Rs.5000/-
c) Multiple inheritence is not supported for interfaces
d) No Error, No output