

Explore

EN ▾ What do you want to learn?

TypeScript

TypeScript is a superset of JavaScript which provides additional features like static typing, module creation, and object-oriented concepts. This course introduces l...More

Start

Learning Progress ?

Status: Completed

Completion Certificate

Click on Generate Certificate button to claim your completion certificate

Generate Certificate

Overview

Contents

Discussions

What you will learn

At the end of this course, the learner will be able to: Create optimized JavaScript code by using a strict type checking concept. Develop reusable code by using enhanced functional programming and Object-Oriented programming features Modularize the application using namespaces/modules. Invoke other JavaScript code/libraries within TypeScript code.

Pre Contents

Authors/Creators

AK

Anitha Kumari K.

VB

Veena Baburai Bhat

RS

Rafi Sneha M C

KK

Khalid Kamal Hussain

Show Curators/Contacts

At a glance

Course

13h 37m

Beginner Level

Free

Infosys Wingspan

EN

TypeScript, TS, ES6, ECMAScript6, JavaScript, TypeScript Basics, Modules and Namespace, Generics, TypeScript Class, TypeScript

Thank you. Your test submitted.

You have cleared this assessment.

Obtained Percentage

Obtained Marks

88 %

22 / 25

Best Attempt Score:88 % on 17-06-2025

[Review Your Attempt](#)

Find error if any in the below code:

```
1. class a{
2.     parent: string = "From Parent";
3. }
4. interface b{
5.     fromInterface: number;
6. }
7. class c implements b extends a {
8.     fromInterface = 20;
9.     para = () => {
10.         console.log(this.parent);
11.     }
12. }
13. new c().para();
```

Warning

This operation is disabled.

Ok

- ☐ Lines 7 and 10 will cause errors
- ☒ Line 7 alone causes error because extends clause must precede implements clause
- ☐ Line 7 alone causes error because we can't use implements clause and extends clause together with a single class
- ☐ Line 10 alone causes error because we need to call using super.parent

What will be the output for the below code?

```
let var1 = 20;  
function fun() {  
    var1 = 21;  
    var a = () => { var1 = 40 };  
    a();  
}  
fun();  
alert(var1);
```

☐ 20

☐ 21

☒ 40

☐ Error cannot redeclare var1

Predict the output for the below code:

```
1. interface a
2. {
3.     name: string;
4. }
5. interface b
6. {
7.     age: number;
8. }
9. interface c extends a, b
10. {
11.     cash: number;
12. }
13. class ab implements a, b
14. {
15.     cash = 2000;
16.     name = "Barry";
17.     age = 20;
18.     public a = () => alert(this.cash + " was the " + this.name);
19. }
20. new ab().a();
```

Warning

This operation is disabled.

Ok

☒ 2000 was the amount Barry had when he was 20

☐ You can't implement more two classes

☐ You have to use extends keyword

☐ There is an error in line 18

For More Solutions: <https://github.com/DevGoyalG/NIET-Infosys-Springboard>

Predict the output for the below code:

```
function fun(str: string): string {  
    return 42 + " " + str;  
}  
  
function fun(num: number): string {  
    return "hello! " + num;  
}  
  
let d: string = "hello!";  
var disp: string = fun(d);  
console.log(disp);
```

☐ 42

☐ 42 hello

☐ hello 42

☒ Error: Duplicate function implementation

Which of the following line/lines will lead to the compilation error/errors?

```
var a: [string, number];
```

1. `a = ["Stephen Strange", null, "Tony Stark", 2, 6];`
2. `a = ["Stephen Strange", undefined, "Tony Stark", 2, 6];`
3. `a = [undefined, "Tony Stark", 2, "Stepher`
4. `a = [null, null, 4, "Tony Stark", 2, 6];`

Warning

This operation is

- ☐ 1, 2 and 3
- ☐ 1, 2 and 4
- ☐ 2, 3 and 4
- ☒ All of the options

Predict the output for the below code:

```
1.    interface a {  
2.        var1: string;  
3.    }  
4.    interface b {  
5.        var2: string;  
6.    }  
7.    function fun1(para1: a): void {  
8.        alert("Inside fun1" + para1);  
9.    }  
10.   function fun2(para2: b): void {  
11.       alert("Inside fun2" + para2);  
12.   }  
13.   var obj = { var1: "for a", var2: "for b"  
14.       fun1(obj);  
15.       fun2(obj);
```

- ☐ Line 14 will get compilation error
- ☐ Line 15 will get compilation error
- ☐ Both 14 and 15 will get compilation error

Choose an appropriate answer in order to execute the below code without any error:

```
function fun(a) {  
    alert("List: " + a);  
}  
fun("Sherlock", "Dr.Watson", "Inspector Lestrade")  
fun("Irene Adler");  
fun();
```

- ☐ a should be a rest parameter of string[] type
- ☐ a should be an Array of strings
- ☒ a can be a rest parameter or an array of strings
- ☐ None of the options

Warning

This operation is disabled.

Ok

What will be the output for the following?

```
enum cost { Blue = 5000, Green = Blue, Red = Blue+2500, White = Green , Black=2000 };  
console.log(cost[5000]);
```

- ☐ Blue
- ☐ Green
- ☒ White
- ☐ Compilation Error : You cannot reference me

Warning

This operation is disabled.

Ok

ation inside another enum declaration.

What will be the output for the below code?

```
class A{  
    printData<T>(data: T): T{  
        if (data == "Barry Allen")  
            alert("Data Received");  
        return data;  
    }  
}  
  
let obj = new A();  
let data:string=obj.printData<string>('Barry Allen')
```

Warning

This operation i

- ☐ Data Received
- ☐ Error : Cannot have generic functions inside a non-generic class
- ☐ Error : Operator '==' cannot be applied to types 'T' and 'string'
- ☒ Both the above mentioned errors will occur

For More Solutions: <https://github.com/DevGoyalG/NIET-Infosys-Springboard>

Predict the output for the below code:

```
let list: any = ["One", "Two", "Three", "Four"];  
list.push(list.splice(-4, 2));  
alert(list);
```

- ☒ Three, Four, One, Two
- ☐ Four, One, Two, Three
- ☐ One, Two, Three, Four
- ☐ Error : Array out of bounds

What will be the output for the following code?

```
1. function fun(contact: number, name?: string, defaultName: string = "Usopp", ...restName: string[])
2.     {
3.         alert("defaultName has: " + defaultName);
4.     }
5.     fun(1234567891, undefined, "Luffy", "Zoro")
```

- ☒ defaultName has: Luffy
- ☐ undefined
- ☐ Zoro
- ☐ Usopp

Warning

This operation is disabled.

Ok

Consider that you have created two TypeScript files as mentioned below:

```
//One.ts  
__1__ function show (userId:number) : number  
{  
  return 'Your id is ' +`${userId}`  
}
```

```
//Two.ts  
__2__ {show} from './One';  
console.log(show(1234));
```

Fill in the blank with the appropriate option to execute the above code.

- ☐ 1 - import , 2 - export
- ☒ 1 - export , 2 - import
- ☐ 1 - exports , 2 - import
- ☐ 1 - imports , 2 - export

Warning

This operation is disallowed

Predict the output for the below code:

```
1.      const alterEgo = "Slade Wilson";
2.      function outerFunction()
3.      {
4.          let alterEgo = "Wade Wilson";
5.          setTimeout(
6.              function () {
7.                  alert(alterEgo);
8.              }, 2000);
9.      }
10.     outerFunction();
```

- ☐ Slade Wilson
- ☒ Wade Wilson
- ☐ Error in line 4
- ☐ Error in line 2

Find which lines cause compilation errors from the below code:

```
1.    class company{
2.        static empId:number=753800;
3.        private _eta: boolean;
4.        constructor(){
5.            this.empId++;
6.            this._eta = false;
7.        }
8.        display(dept:boolean):boolean
9.        {
10.            this._eta=dept;
11.            return this._eta;
12.        }
13.    }
14.    var emp1=new company();
15.    emp1.display(false);
16.    alert(emp1);
17.    let emp2 = new company();
18.    alert(emp2.empId);
```

Warning

This operation

☐ Line 5 alone

☐ Line 11 alone

☒ Lines 5 and 18

☐ Lines 5, 11 and 18

For More Solutions: <https://github.com/DevGoyalG/NIET-Infosys-Springboard>

Predict the output for the below code.

```
interface a {  
    a: string;  
    abstract fun(): string;  
}  
class b implements a {  
    a: string;  
    constructor() {  
        this.a = "Success!"  
    }  
    fun() {  
        return this.a;  
    }  
}  
let obj = new b;  
console.log(obj.fun());
```

Warning

This operation is di

- ☐ Success!
- ☐ Cannot implement interface with an abstract function
- ☐ Error : class b has to define function fun inside the constructor
- ☒ Error : abstract member cannot appear on a type member

For More Solutions: <https://github.com/DeyGoyalG/NIET-Infosys-Springboard>

Predict the output for the below code:

```
class Friends<T>{  
    characterList: Array<T> = [];  
    addCharacter(newCharacterList: Array<T>): void {  
        this.characterList = newCharacterList;  
        alert(characterList);  
    }  
}  
  
let show = new Friends<string>();  
let characterList: Array<string> = ["Chandler", "Monica", "Joe", "Rachel", "Ross", "Phoebe", "Joey", "Joey"];  
show.addCharacter(characterList);  
  
let show2 = new Friends<number>();  
let seasons: Array<number> = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];  
show2.addCharacter(seasons);
```

Warning

This operation

- ☒ ['Chandler', 'Monica', 'Joe', 'Rachel', 'Ross']
['Chandler', 'Monica', 'Joe', 'Rachel', 'Ross']
- ☐ ['Chandler', 'Monica', 'Joe', 'Rachel', 'Ross']
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
- ☐ [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

For More Solutions: <https://github.com/DevGoyalG/NIET-Infosys-Springboard>

Consider you are using a tsconfig.json file. The ts files are stored in a folder named ts. tsconfig.json is stored outside the ts folder. Which compiler option is best suited to read the ts files from the folder while transpiling?

- ☒ rootDir
- ☐ outDir
- ☐ sourceDir
- ☐ None of the options

Warning

This operation is disabled.

What will be the output for the below code?

```
let ceoList: string[]=[];  
ceoList["XYZ"] = "Tim";  
ceoList["ABC"] = "Tom";  
ceoList["TEST"] = "Joe";  
ceoList["TEXT"]="John";  
alert("Length : " + ceoList.length);  
alert("Ceo of XYZ: " + ceoList["XYZ"]);
```

- ☐ Length : 4 Ceo of XYZ: Tim
- ☒ Length : 0 Ceo of XYZ: Tim
- ☐ Length : 0 Ceo of XYZ: Undefined

Predict the output of the below-given TypeScript code.

```
const students : any [] = [  
  { studentName: 'Tim', studentId: 1034},  
  { studentName: 'Jack', studentId: 2345},  
  { studentName: 'Jim', studentId: 3456}  
]  
console.log(students.splice(1,1));
```

Warning

This oper

- ☒ [{ studentName: 'Jack', studentId: 2345}]
- ☐ [{ studentName: 'Jim', studentId: 3456}]
- ☐ [{ studentName: 'Jack', studentId: 2345},
{ studentName: 'Jim', studentId: 3456}]
- ☐ [{ studentName: 'Tim', studentId: 1034},
{ studentName: 'Jack', studentId: 2345},
{ studentName: 'Jim', studentId: 3456}]

Select the statement that helps a TypeScript developer to understand the purpose of Decorators?

- ☐ Helps in declaring method with a metadata.
- ☐ Helps in declaring lexical scope variables.
- ☒ Helps in declaring classes with predefined metadata.
- ☐ Helps in declaring interface with a metadata.

Warning

This operation is disabled.

What is the data type used in TypeScript to declare a variable that does not hold any value in the entire application?

- ☐ number
- ☐ string
- ☒ never
- ☐ any

Warning

This operation is disabled.

Predict the output for the below code:

```
//module.ts:
class Utility {
    setId(name: string, id: number): string {
        return name + " " + id;
    }
}

export const identity: string = "EMPLOYEE";
export { Utility as mainUtility };

//file.ts:
import * as mainutil from "./module";
let util = new mainutil.mainUtility();
let emp = util.setId("JA$", 753886);
console.log(`${emp} is an ${mainutil.identity}`);
```

- ☒ JA\$ 753886 is an EMPLOYEE
- ☐ Error : mainutil is not defined in module.ts
- ☐ Error : mainUtility is not defined in module.ts
- ☐ Error : cannot use export block for classes

Predict the output of the below code snippet.

```
class company {  
    private _empName: string = "Jack";  
    private _empDept: string = "Finacle";  
    constructor(empName: string, empDept: string)  
    {  
        this._empName = empName;  
        this._empDept = empDept;  
    }  
    display(): void {  
        console.log(this._empName + " works for " + this._empDept);  
    }  
}  
  
var emp1 = new company("Tom", "Training");  
emp1.display();
```

- ☒ Tom works for Training unit
- ☐ Jack works for Training unit
- ☐ Tom works for Finacle unit
- ☐ Jack works for Finacle unit

Consider the below-given TypeScript code that populates studentId value on the default console. Can you help in updating this code to implement the same concept using rest parameter usage.

```
function showDetails(studentName:string, studentId:number, studentRank:number):number
{
    return studentId;
}
let studentInfo:number = showDetails("Tim", 1234,
console.log(studentInfo);
```

Warning

This operation is disabled.

Ok

- ☐ replace function declaration as:
function showDetails(studentName:string, ..
- ☐ replace return value of function as:
return studentId[0];
- ☒ Both the mentioned changes must be implemented.
- ☐ Both the given options do not help in implementing this requirement.

Predict the output of the below-given code snippet.

```
function add(num1 = 200, num2?: number) {  
    if (num1) {  
        return num1 + num2;  
    }  
    return num2;  
}  
console.log(add(3, 5) + add(undefined, 5));
```

- ☐ NaN
- ☒ 213
- ☐ Runtime Error