Function initialization and code

Function eg()

{

// body

}

Calling function

eg();

Function eg(a:number, b:string)

name(10,”Gopal”); // correct

name(10); // incorrect number of argument

name(“Gopal”, 10) // parameter is number string given in argument

**Function eg( a?:string, b:number){**

**}**

**eg();**

// Error -> a required parameter can not follow an optional parameter.

Means if you define any optional parameter then after that all parameter should be optional, we can not define required parameter after optional.

We can assign default value to parameter

**Function eg(a:string, b=20)**

**{**

**}**

By default value of b will be 20 if not passing any value while calling.

If we assign any value in parameter then next parameters should have default values or it should be optional.

**Function eg(a:string, b=20, c:string)**

**Function eg(a:string, b=20, c?:string)** // should be like this.

We can pass variable number of argumesnts (..var:type[])

**Function eg (author:string, …books:string[]){}**

We can not define multiple variable parameters in one function.

It means if we are defining any variable length parameter then it should be last parameter of the function.

“ A rest parameter must be last in a parameter list.” - error by compiler.

Code:

function someEg10(a:number)

{

if(a>10)

return a\*a;

return "Useless";

}

function eg10()

{

let a:number;

let b:string;

a = someEg10(5);

b = someEg10(20);

console.log(a);

console.log(b);

}

Have Error:

Type 'number | "Useless"' is not assignable to type 'number'.

Type 'string' is not assignable to type 'number'.

Solution:

function someEg11(a:number)

{

if(a>10)

return a\*a;

return "Useless";

}

function eg11()

{

let a:any; // change here

let b:any; // change here

a = someEg11(5);

b = someEg11(20);

console.log(a);

console.log(b);

}

Pointer to function.

Declaring a pointer which is capable of storing address of the function.

Function egOne(a:number):number

Function egTwo():string

egOne(10)

egTwo();

Let aa:(a:number)=>number;

Let bb:()=>string;

aa(10); // same output as egOne

bb(); // same output as egTwo