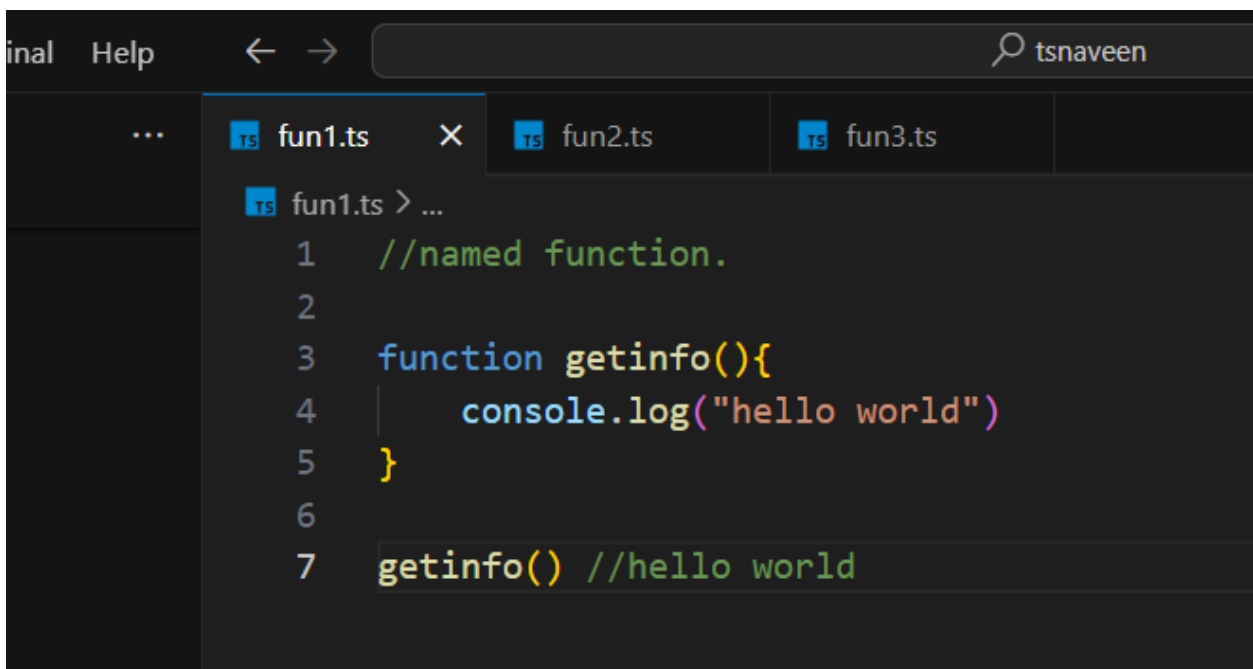
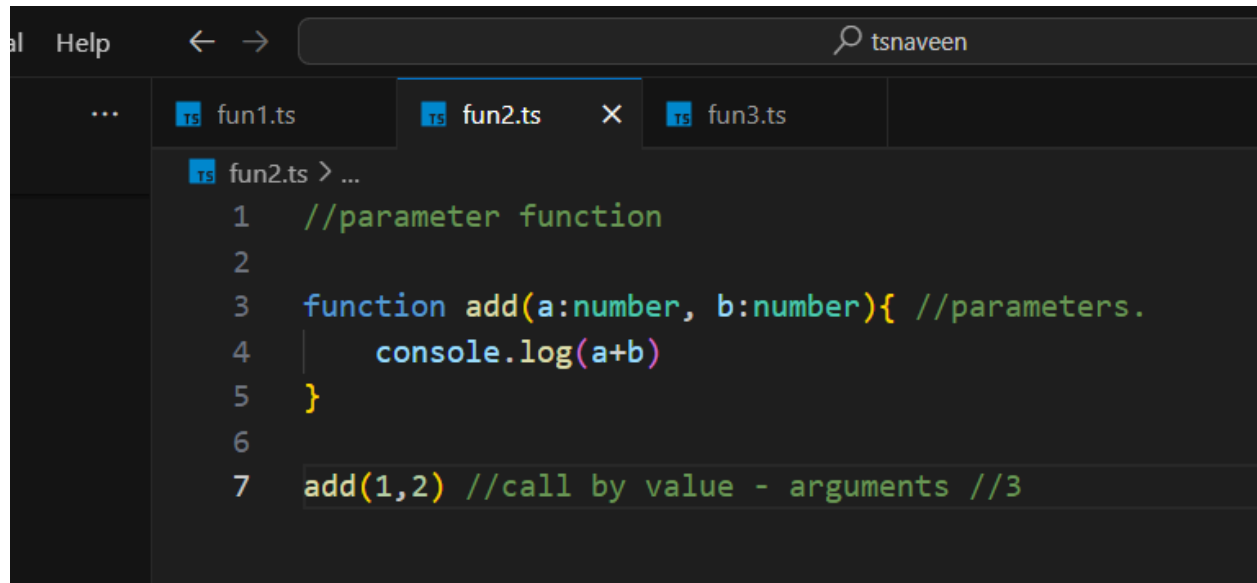


Parameterised function



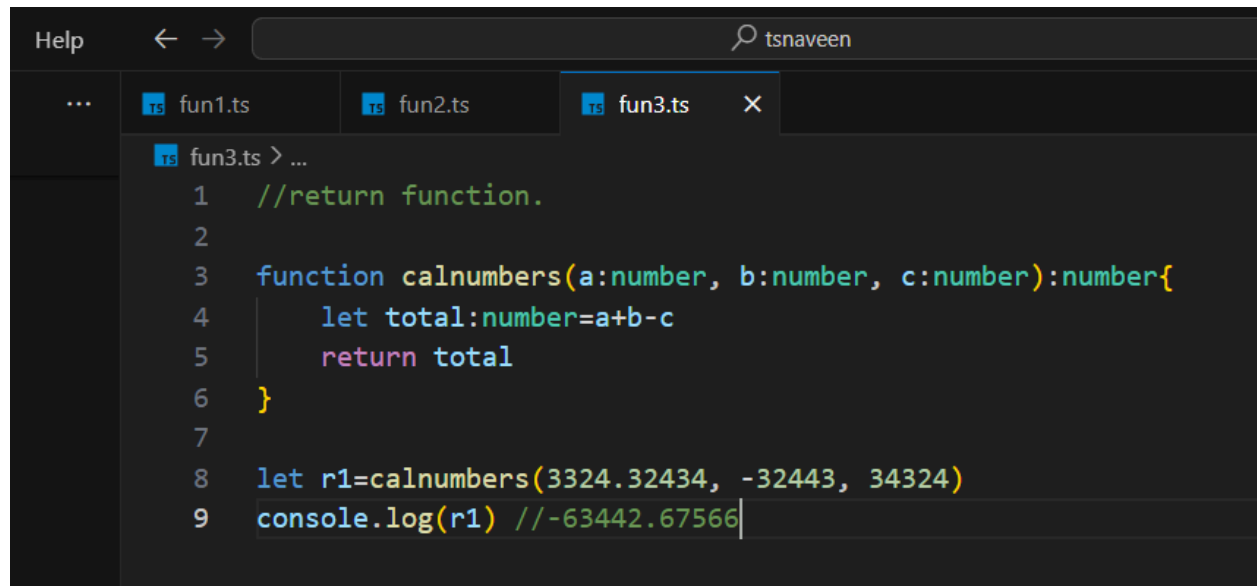
Return function



```
al  Help  < >  🔍 tsnaven

...  TS fun1.ts  TS fun2.ts  X  TS fun3.ts

TS fun2.ts > ...
1  //parameter function
2
3  function add(a:number, b:number){ //parameters.
4      console.log(a+b)
5  }
6
7  add(1,2) //call by value - arguments //3
```



```
Help  < >  🔍 tsnaven

...  TS fun1.ts  TS fun2.ts  TS fun3.ts  X

TS fun3.ts > ...
1  //return function.
2
3  function calnumbers(a:number, b:number, c:number):number{
4      let total:number=a+b-c
5      return total
6  }
7
8  let r1=calnumbers(3324.32434, -32443, 34324)
9  console.log(r1) //-63442.67566
```

Boolean

```
tsnaveen
fun4.ts 1 x fun5.ts 1 fun6.ts 1 fun7.ts 1 fun8.ts
fun4.ts > isuseractive
1 //boolean function.
2
3 function isuseractive(username:string):boolean{
4     if(username==='jack'){
5         console.log('titanic')
6         return true
7     }else if(username==='honey'){
8         console.log('singh')
9         return false
10    }else{
11        console.log('daruwala')
12        return false
13    }
14 }
15
16 let u1=isuseractive('karan')
```

```
15
16 let u1=isuseractive('karan')
17 console.log(u1)
18
19 // daruwala
20 // false
```

Calling function directly inside if-

```
Help  < ->  tsnaven

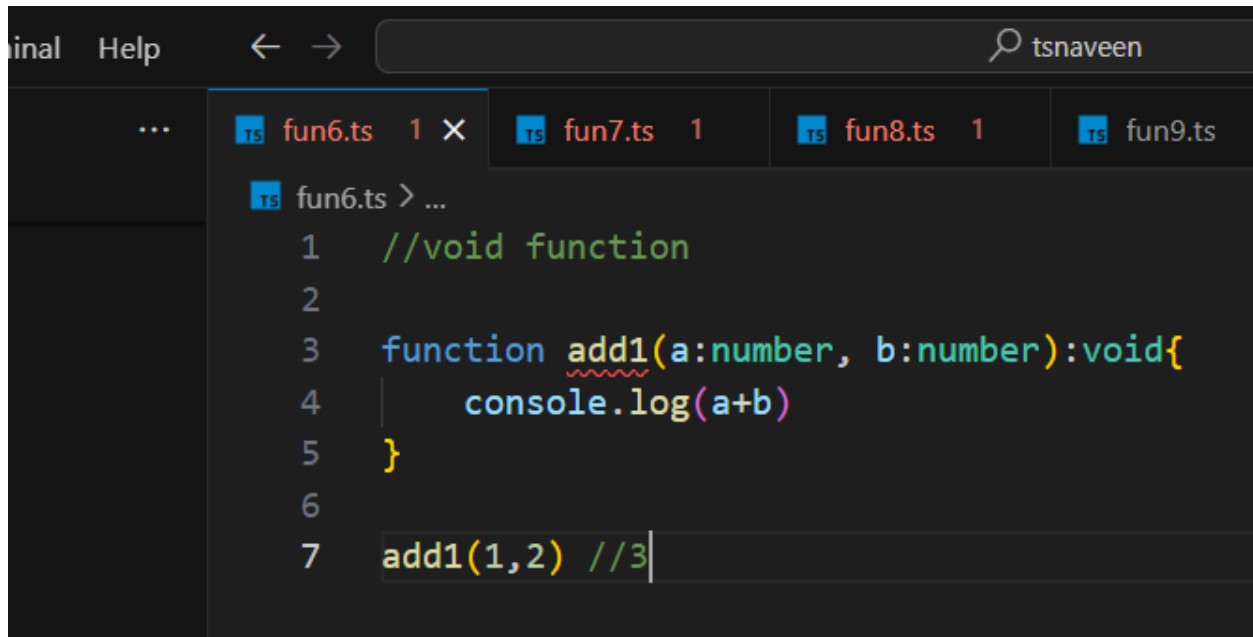
...  fun5.ts  fun6.ts 1  fun7.ts 1  fun8.ts 1  fun9.ts

fun5.ts > ...
1 //call function directly inside if.
2 //see when we write something then the return is not printed.
3
4 //boolean function.
5
6 function isuseractive(username:string):boolean{
7     if(username==='jack'){
8         console.log('titanic')
9         return true
10    }else if(username==='honey'){
11        console.log('singh')
12        return false
13    }else{
14        console.log('daruwala')
15        return false
16    }
17 }
```

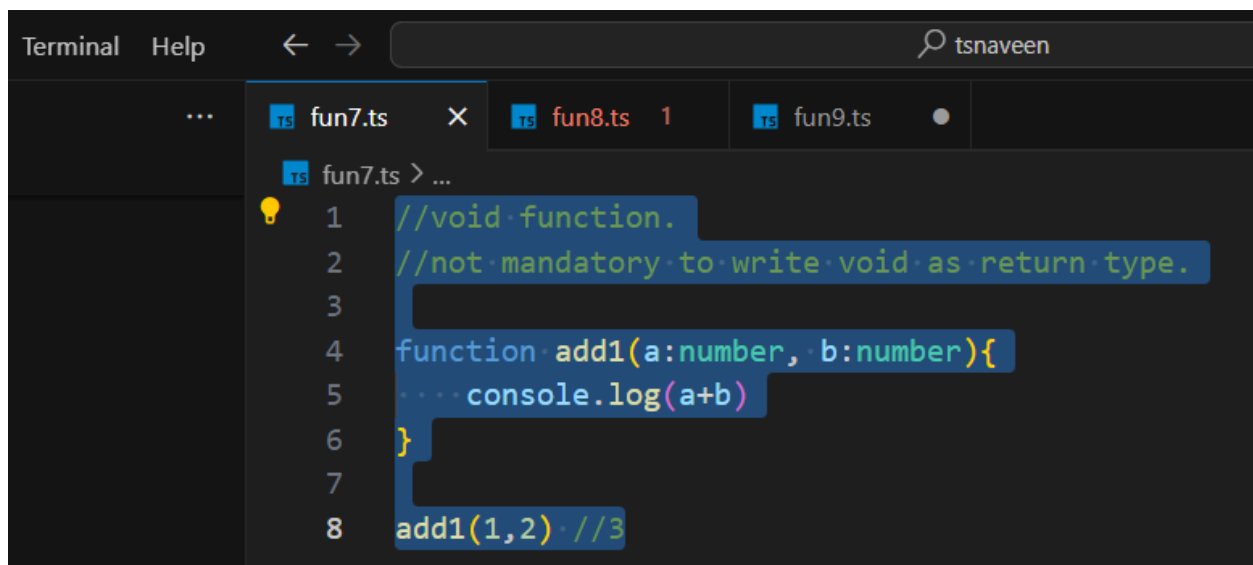
```
17 }
18
19 if(isuseractive('jacky')){
20     console.log('login with jacky')
21 }
22
23 //daruwala
24
25
26 if(isuseractive('jacky')){
27 }
28
29 //daruwala
```

Void function

Not mandatory to write void.

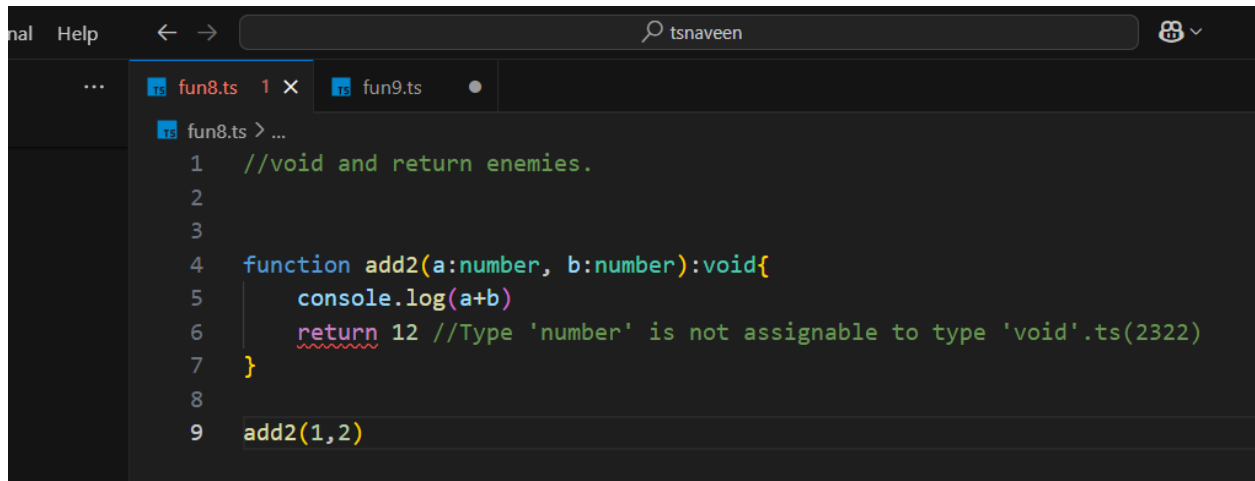


```
1 //void function
2
3 function add1(a:number, b:number):void{
4     console.log(a+b)
5 }
6
7 add1(1,2) //3
```



```
1 //void function.
2 //not mandatory to write void as return type.
3
4 function add1(a:number, b:number){
5     console.log(a+b)
6 }
7
8 add1(1,2) //3
```

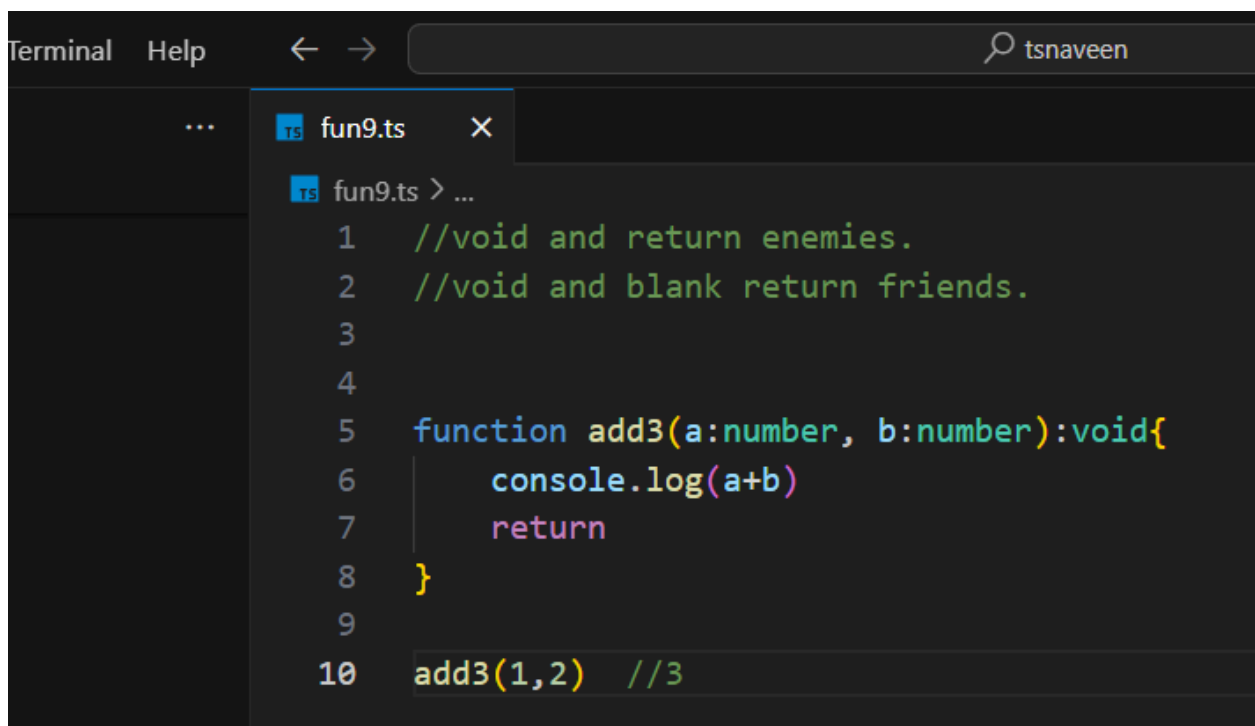
Void and return enemies



The screenshot shows the VS Code editor with a file named `fun8.ts`. The code defines a function `add2` with parameters `a:number` and `b:number`, and a return type of `void`. Inside the function, `console.log(a+b)` is called, and then `return 12` is executed. A red squiggly line under `12` indicates a TypeScript error: `//Type 'number' is not assignable to type 'void'.ts(2322)`. Below the function, `add2(1,2)` is called.

```
1 //void and return enemies.
2
3
4 function add2(a:number, b:number):void{
5     console.log(a+b)
6     return 12 //Type 'number' is not assignable to type 'void'.ts(2322)
7 }
8
9 add2(1,2)
```

Void and blank return



The screenshot shows the VS Code editor with a file named `fun9.ts`. The code defines a function `add3` with parameters `a:number` and `b:number`, and a return type of `void`. Inside the function, `console.log(a+b)` is called, and then `return` is executed without a value. Below the function, `add3(1,2)` is called, followed by a comment `//3`.

```
1 //void and return enemies.
2 //void and blank return friends.
3
4
5 function add3(a:number, b:number):void{
6     console.log(a+b)
7     return
8 }
9
10 add3(1,2) //3
```

TS functions.ts x

TS functions.ts > ...

```
44 //2. Anonymous function: without name function -- store in a variable:
45 //call it using the variable name only.
46
47 let info = function(){
48     console.log("hello ts");
49 }
50
51 info();
```

Terminal Help

TS fun10.ts x

TS fun11.ts

TS fun10.ts > ...

```
1 //anonymous function.
2 //without name.
3
4 let i1=function(){
5     console.log("hello")
6 }
7
8 i1() //hello
```

run Terminal Help

tsnaveen

fun10.ts fun11.ts

fun11.ts > ...

```
1
2 let s1=function(a:number, b:number):number{
3     return a+b
4 }
5
6 let s2=s1(1,-32434.32434)
7 console.log(s2) //-32433.32434
8
9
10 let s3:number=s1(1,-32434.32434)
11 console.log(s3) //-32433.32434
```