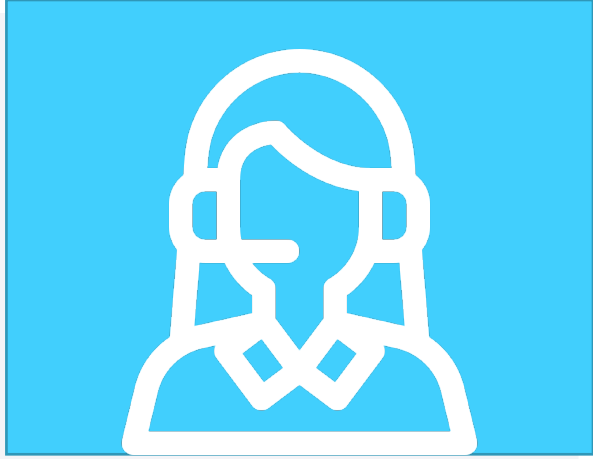
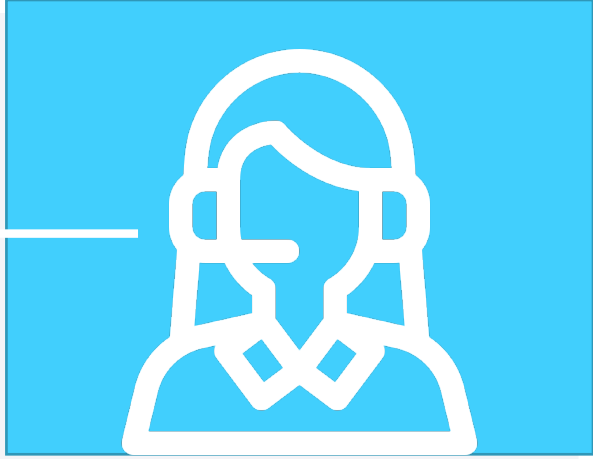


Functions

Content



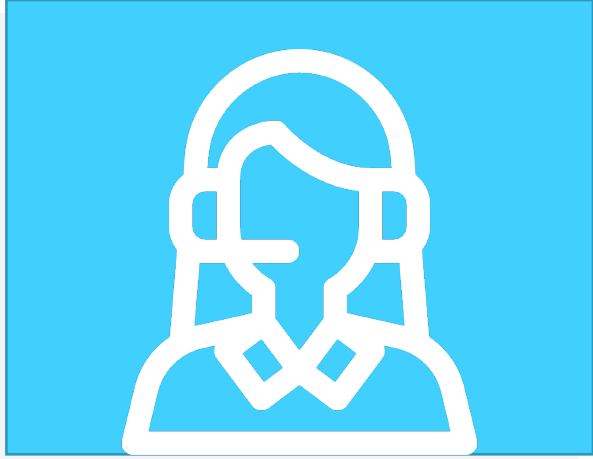
01 What is a function?

02 Declaration

03 Parameters

04 Return value

05 Reference and Value

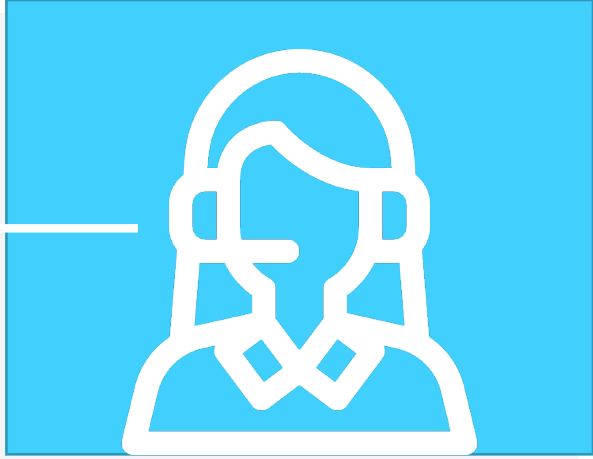


01 Functions

What is a function

Sirma

What is a function?



- A named piece of code that we can call
- Example function definition:

```
function printHello() {
    console.log("Hello");
}
```

Function named printHello

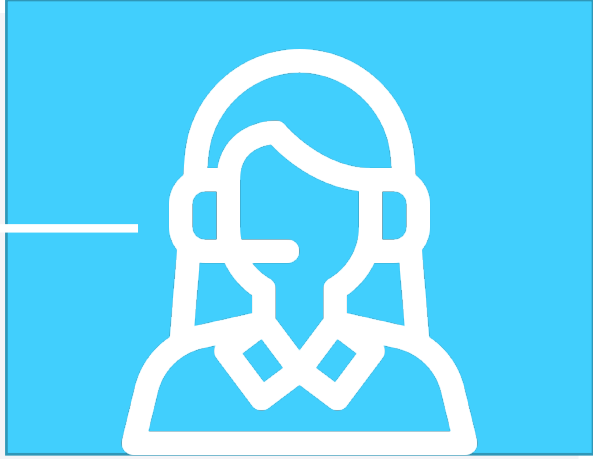
Function body

- We can call the function repeatedly with:

```
printHello();
printHello();
```

Sirma

Why do we use functions?



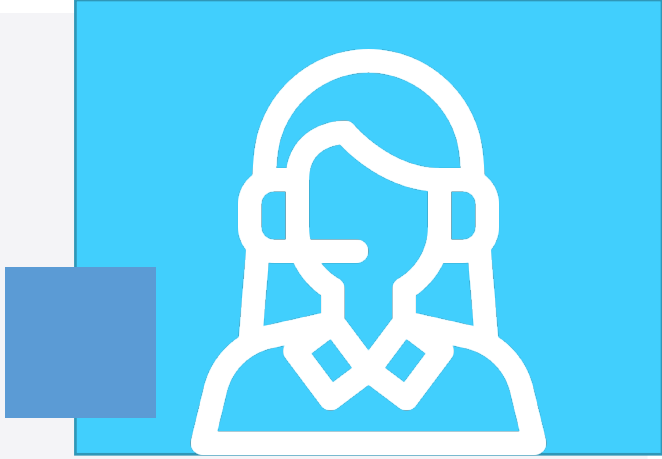
- Easier code management
  - We break down a problem into small steps
  - Better code organization
  - Higher readability
  - Easier to understand code
- We avoid repeating code
  - We're improving code support
- Code reusability
  - We can call the method multiple times

Functions that do not return a value

- Execute the code between the curly braces
- They do not return a result

```
function printHello() {
   console.log("Hello");
}
```

```
function main () {
   console.log("Hello");
}
```



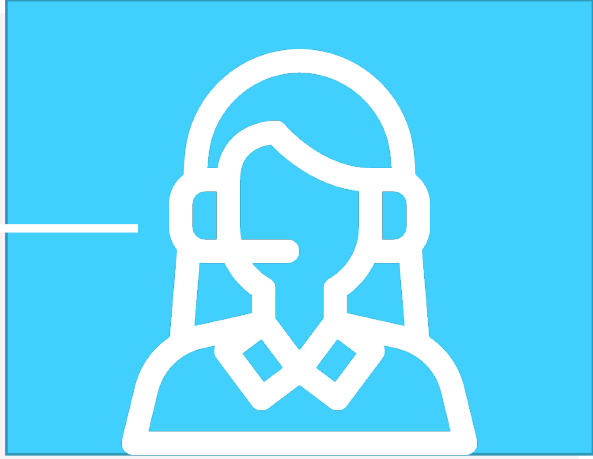
02

Declaring a function

Calling functions

Sirma

Declaring a function



- Methods are declared using the function keyword .
- Variables in a function are local

```
function printText(text) {
   console.log(text);
}
```

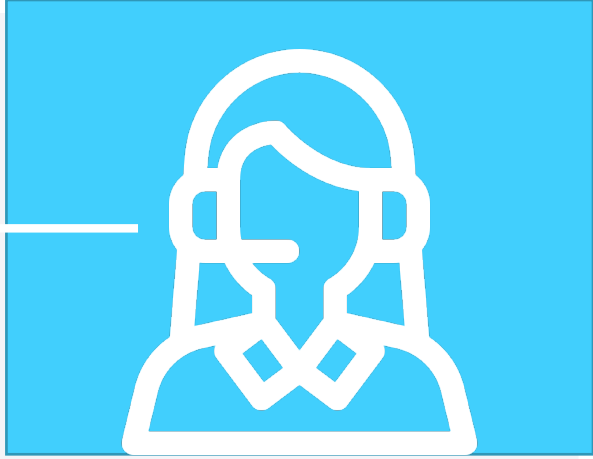
Name

Parameters

Body

Sirma

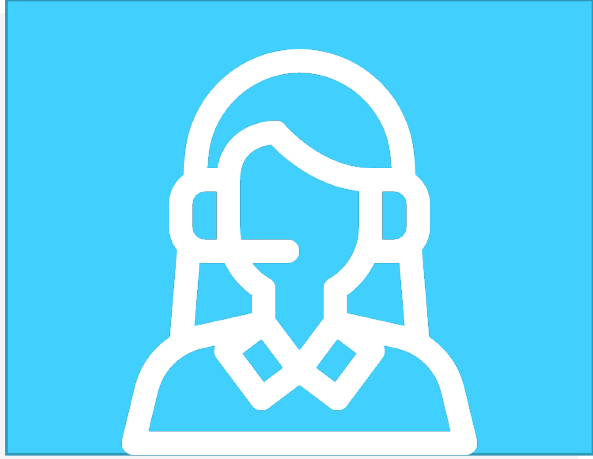
Calling a function



- After declaring a function, we can call it multiple times

```
function printLine () {
  console.log( "----------" );
}
```

```
function resolve() {
  printLine( );
  printLine( );
}
```

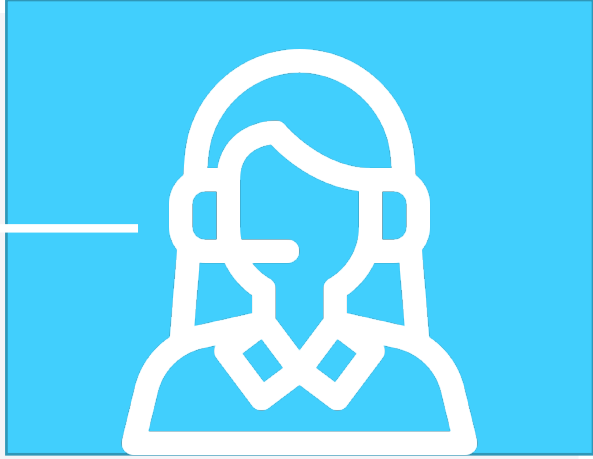


03 Parameters

Passing values

Sirma

Parameters



- Function can take parameters:

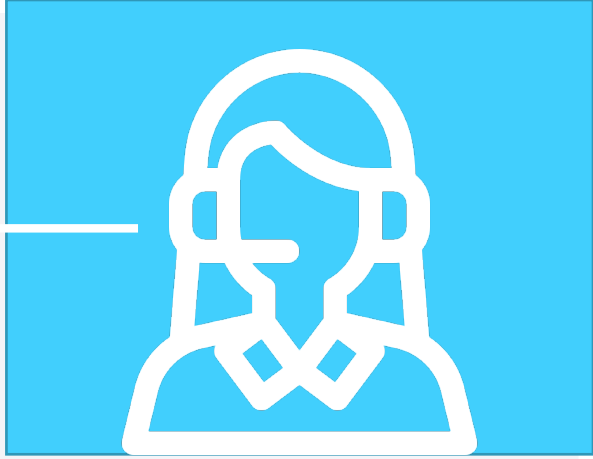
```
function printNumbers( start, end) {
    for ( let i = start; i <= end; i ++)
        console.log(`${i}`);
}
```

- We call the function, passing the appropriate arguments:

```
printNumbers ( 1 , 5 );
printNumbers (10, 100);
```

Sirma

Parameters



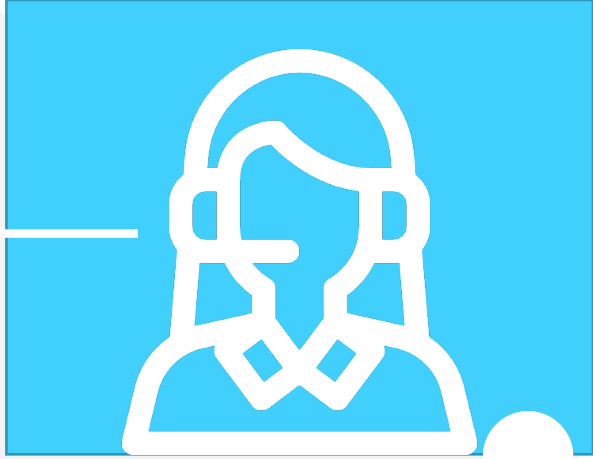
- 0 or more parameters
- Each parameter has a name

Multiple parameters

Name

```
function printStudent ( name , age, grade) {
    console.log( `Student: ${name} - ${age} - ${grade}` );
}
```

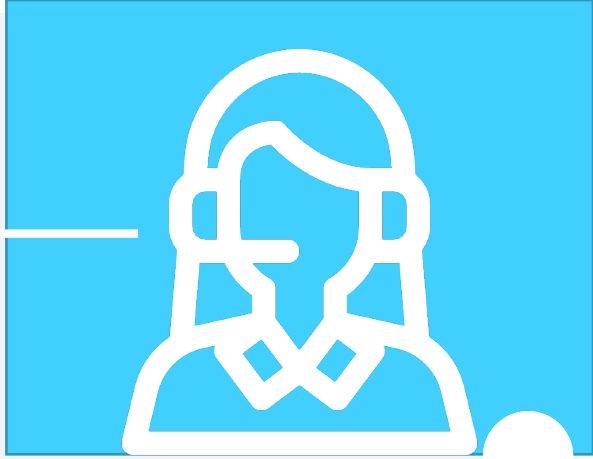
Number sign



Write a function that prints the sign of an integer, which receives as an argument:

`2`	➡	`positive`
`-5`	➡	`negative`
`0`	➡	`zero`

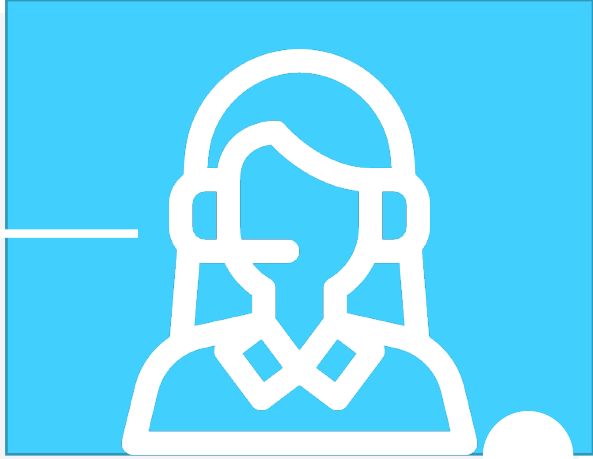
Assessment



Write a function that, given a real number score, prints the score in words:

- 2.00 - 2.99 - "Fail"
- 3.00 - 3.49 - "Poor"
- 3.50 - 4.49 - "Good"
- 4.50 - 5.49 - "Very good"
- 5.50 - 6.00 - "Excellent"

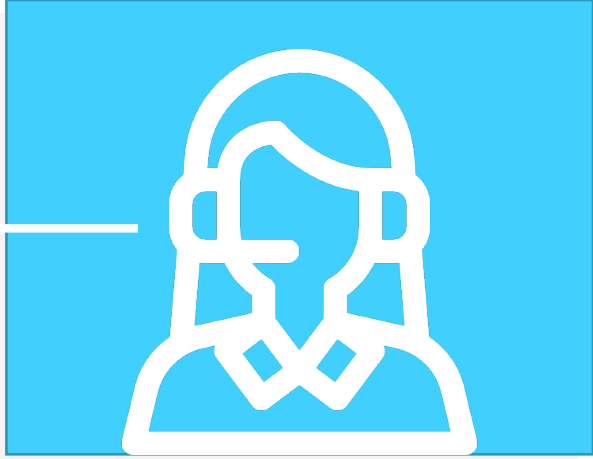
Assessment



```
function printGrade (grade) {
   if (grade >= 2 && grade <= 2.99){
      console.log( "Fail" );
   } else if {
     // TODO
}
```

Sirma

Default value



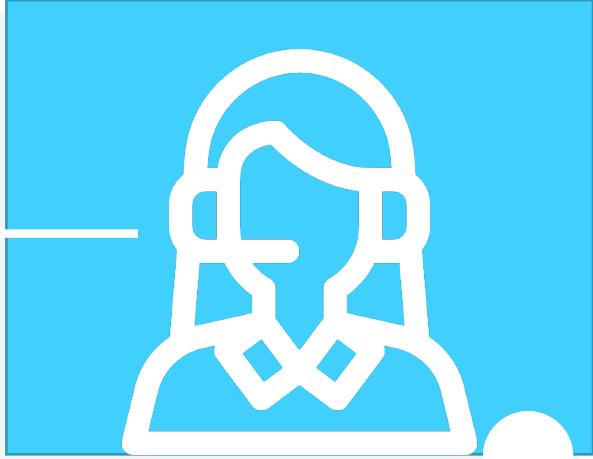
- We can set an initial value to the parameters:

```
function printNumbers (start = 1 , end = 10 ){
    for ( let i=start; i<=end; i++)
        console.log(i);
}
```

- We can call the function without passing any arguments:

```
printNumbers ( 1 , 5 );
printNumbers ();
```

Triangle



Write a function that prints the triangle shape as shown:

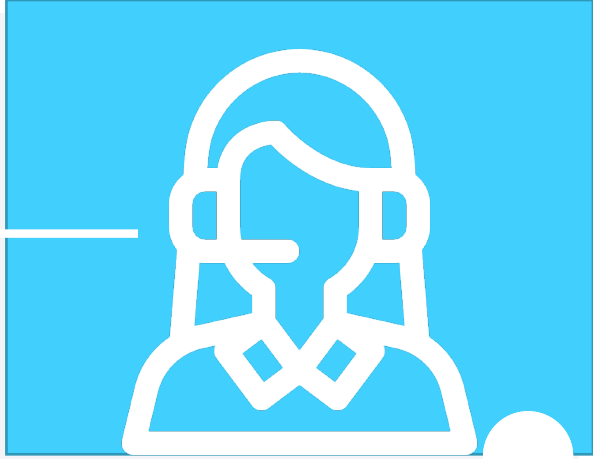
3

```
1
1 2
1 2 3
1 2
1
```

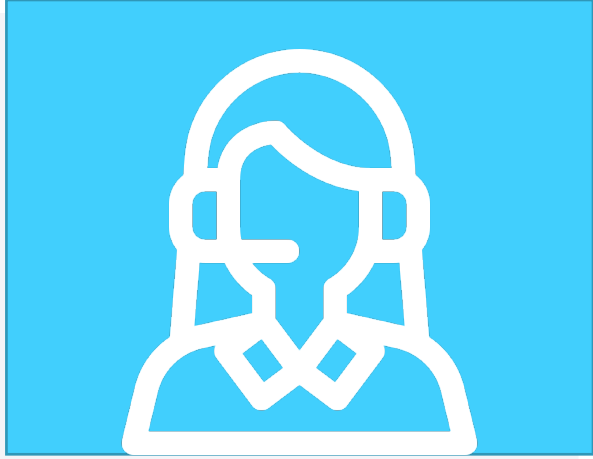
4

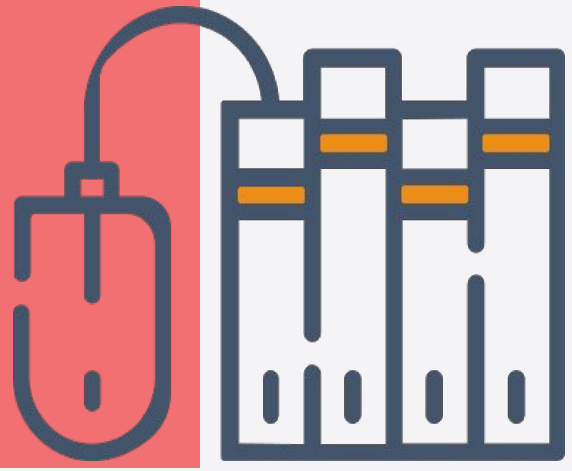
```
1
1 2
1 2 3
1 2 3 4
1 2 3
1 2
1
```

Triangle - Guidelines



- Create a function that prints only 1 row of the figure: printLine()
- Create a function printTriangle , which prints the entire shape by calling function printLine();

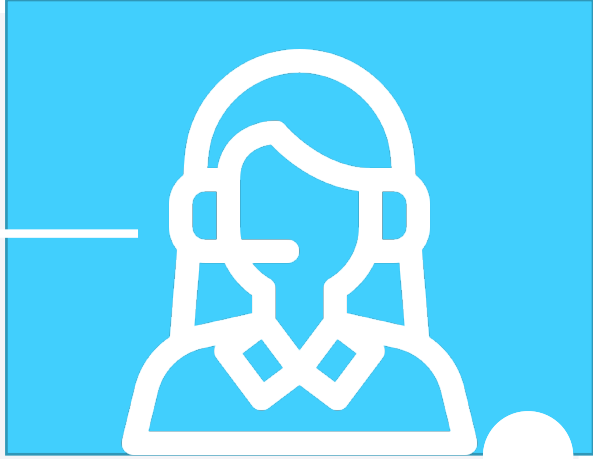




04

Return value

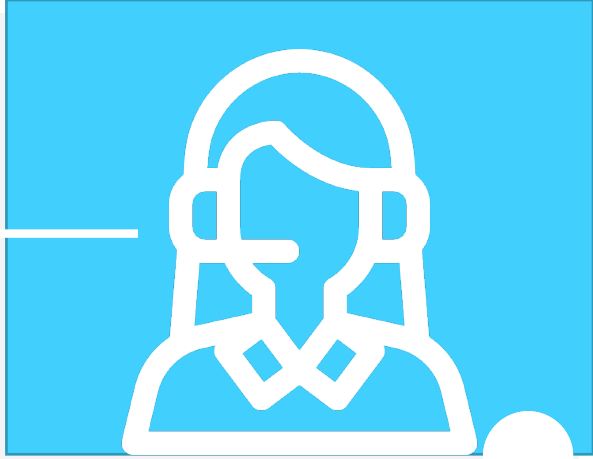
Return



- The keyword **return** terminates the execution of the function
- Returns the specified value back to the function that called it

```
function getFullName( firstName, lastName ) {

    return firstName + " " + lastName ;

}


let fullName = readFullName(" John","Smith ");

console.log(fullName) //John Smith
```

Sample usage

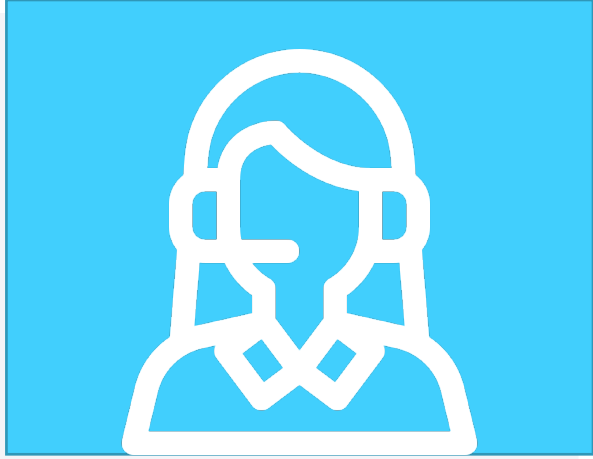


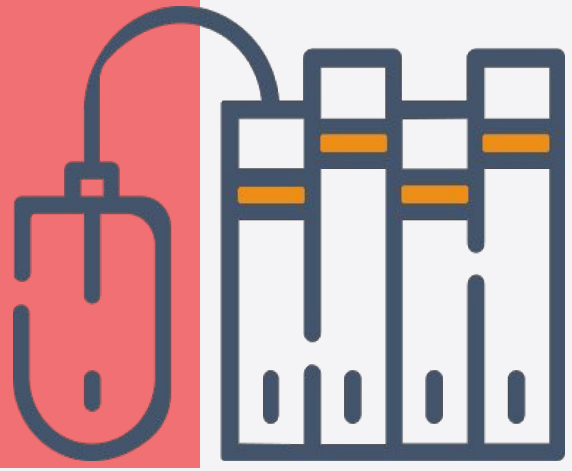
- We check if an index in an array is valid:

```
function isIndexValid(arr, index) {
    if (index < 0 || index >= arr.length ){
        return false;
    } else {
        return true;
    }
}
```

- Whether the student passed the exam :

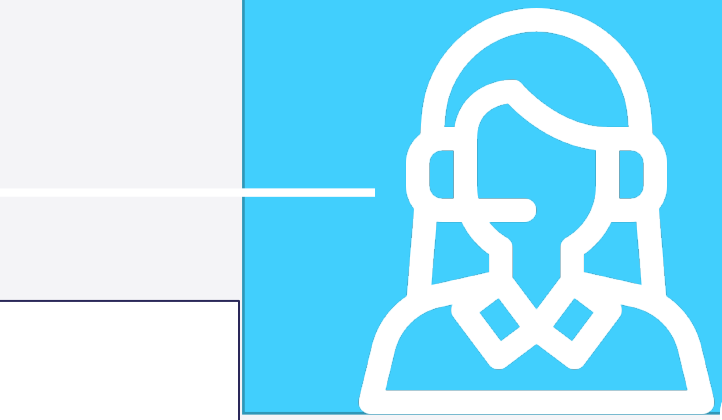
```
function pass(grade) {
    return grade >= 3;
}
```

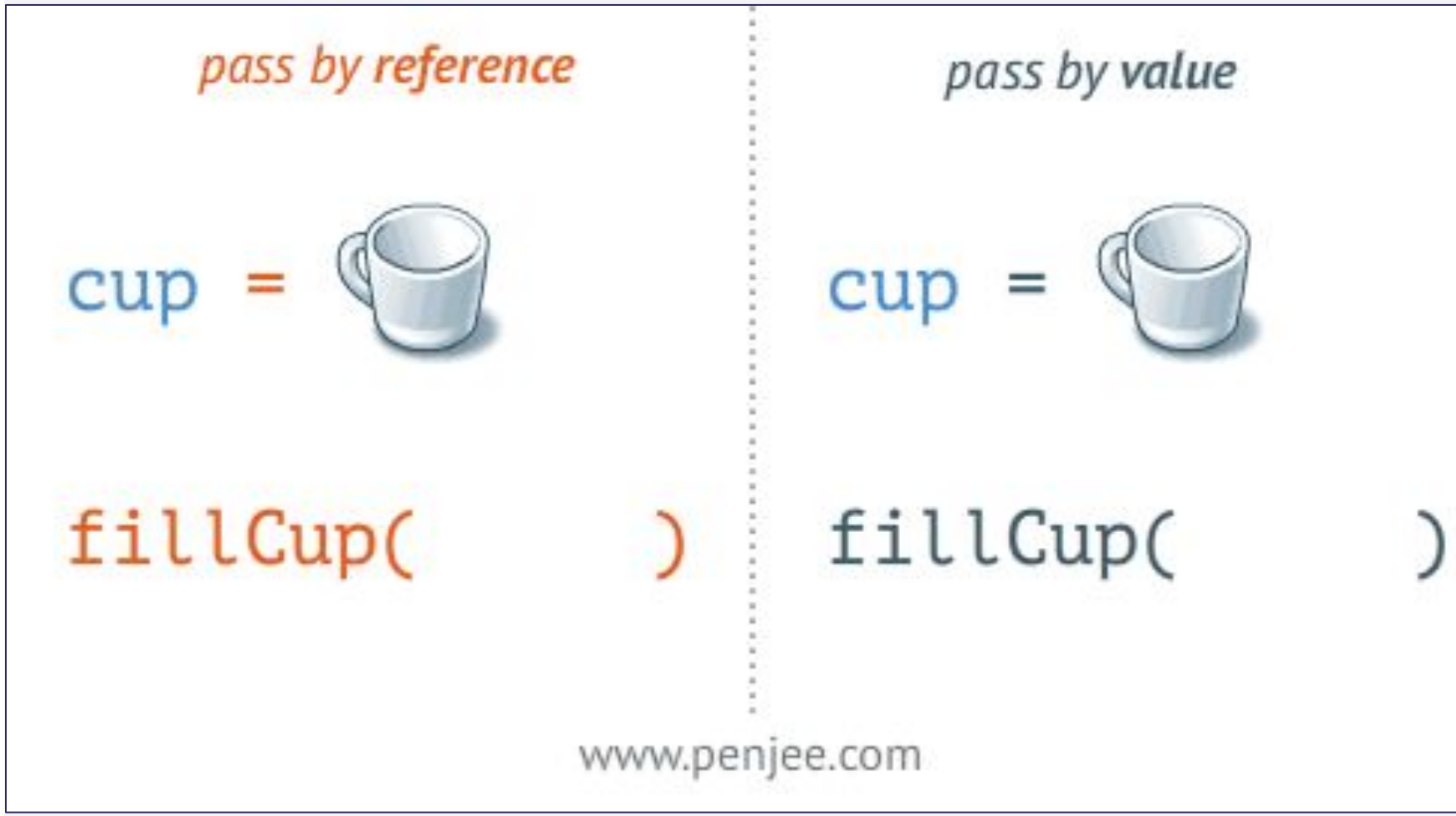




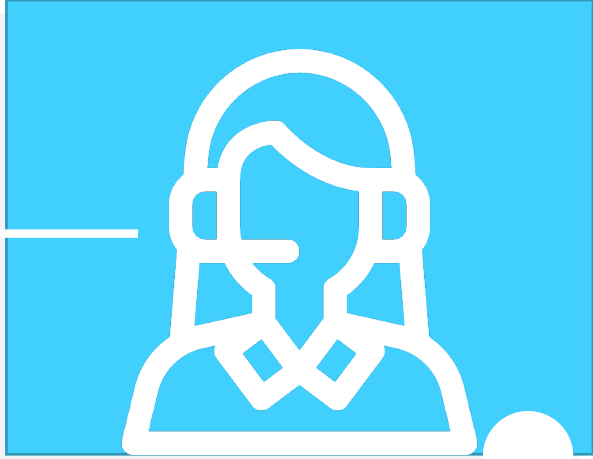
05
Reference

Reference and Value



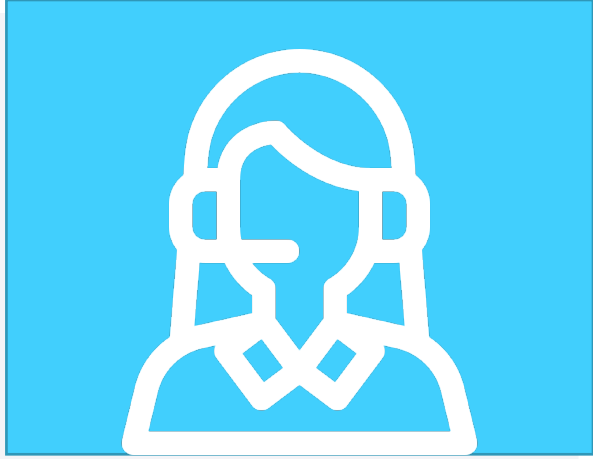


Reference types



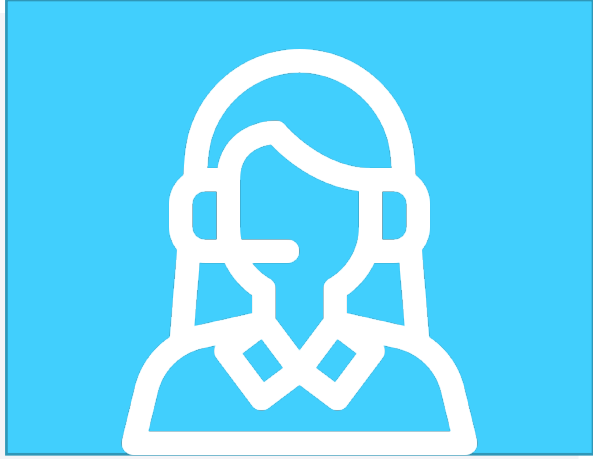
```
function changeName (names) {
    names[0] = "Ivan" ;
}

function resolve() {
    let names = [ "George" , "Peter" ];
    changeName(names);
    console.log(names[0]);
}
```

- Reference types: arrays, objects
- They keep a reference to their value



Summary

- Functions break a problem into small parts
- They have definition and body
- They are called by their name + ()
- They can receive value
- They can return a value



Thank you!