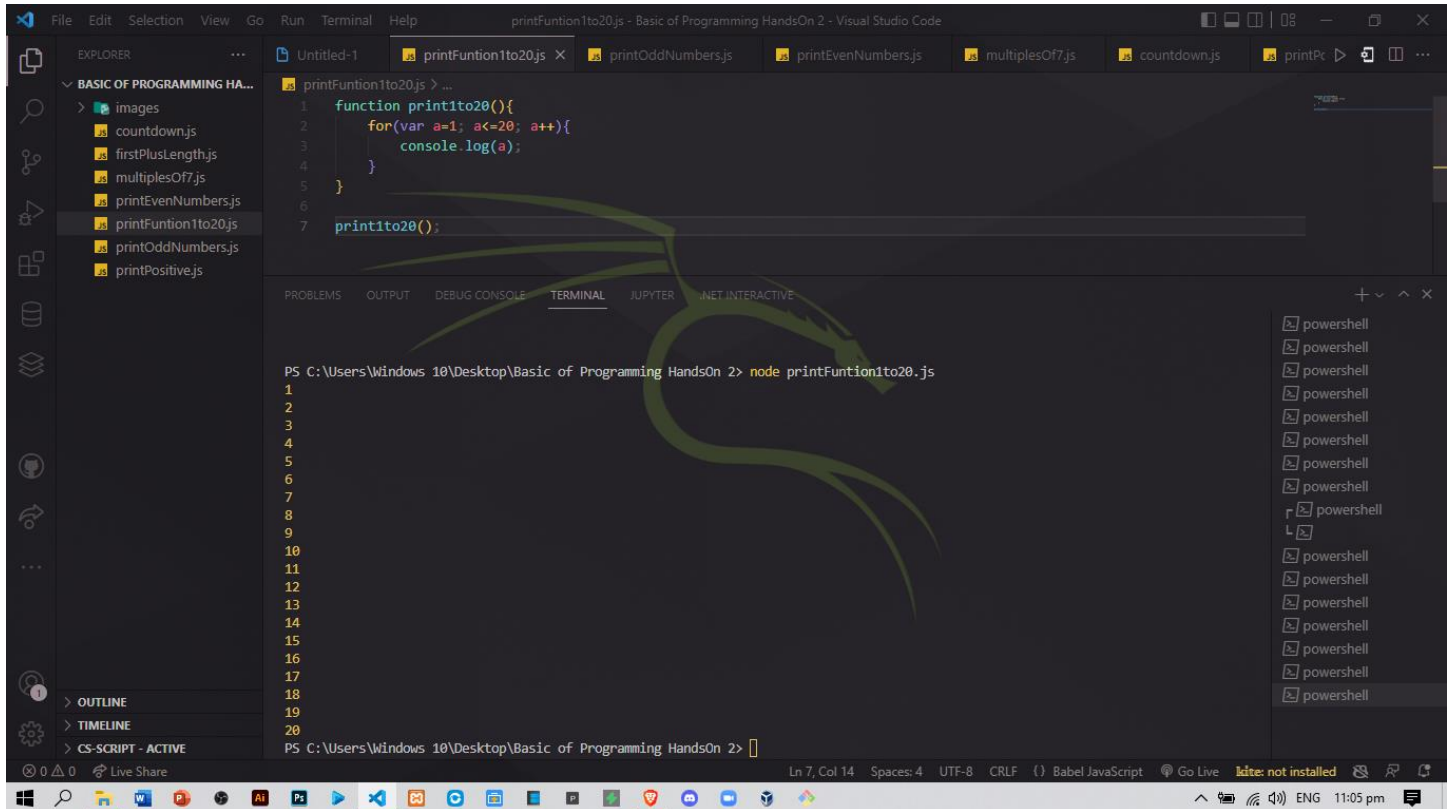


JavaScript HandsOn 2

1. Create a function that prints/logs all the integers from 1 to 20.

Test Cases (0/1)

- print1to20() to log 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20



The screenshot shows the Visual Studio Code editor with a file named `printFunction1to20.js` open. The code defines a function `print1to20()` that uses a `for` loop to iterate from 1 to 20, logging each number to the console. The function is then called. The terminal window at the bottom shows the command `node printFunction1to20.js` being executed, resulting in the output of numbers 1 through 20, each on a new line. The Explorer sidebar on the left shows a project structure with various JavaScript files. The bottom status bar indicates the file is using UTF-8 encoding and CRLF line endings.

```
1 function print1to20(){
2   for(var a=1; a<=20; a++){
3     console.log(a);
4   }
5 }
6
7 print1to20();
```

```
PS C:\Users\Windows 10\Desktop\Basic of Programming HandsOn 2> node printFunction1to20.js
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
PS C:\Users\Windows 10\Desktop\Basic of Programming HandsOn 2>
```

2. Create a function that prints/logs all the odd numbers from 3 to 20.

Test Cases (0/1)

- printOdd3to20() to log 3 5 7 9 11 13 15 17 19

The screenshot shows the Visual Studio Code editor with the file `printOddNumbers.js` open. The code defines a function `printOdd3to20(firstNum, lastNum)` that uses a `for` loop to iterate from `firstNum` to `lastNum`. Inside the loop, it checks if the current number is odd (`oddNum % 2 !== 0`) and logs it to the console. The function is then called with `printOdd3to20(3, 20);`. The terminal shows the output of the command `node printOddNumbers.js`, displaying the odd numbers 3, 5, 7, 9, 11, 13, 15, 17, and 19 on separate lines. The Explorer sidebar on the left lists several other JavaScript files under the 'BASIC OF PROGRAMMING HANDS ON 2' folder.

```
function printOdd3to20(firstNum, lastNum){
  for(oddNum = firstNum; oddNum <= lastNum; oddNum++){
    if(oddNum % 2 !== 0){
      console.log(oddNum);
    }
  }
}

printOdd3to20(3, 20);
```

```
PS C:\Users\Windows 10\Desktop\Basic of Programming HandsOn 2> node printOddNumbers.js
3
5
7
9
11
13
15
17
19
PS C:\Users\Windows 10\Desktop\Basic of Programming HandsOn 2>
```

3. Create a function that prints/logs all the even numbers from 4 to 22.

Test Cases (0/1)

- `printEven4to22()` to log 4 6 8 10 12 14 16 18 20 22

The screenshot shows the Visual Studio Code editor with the file `printEvenNumbers.js` open. The code defines a function `printEven4to22()` that uses a `for` loop to iterate from `a=4` to `a<=22`. Inside the loop, it checks if the current number is even (`a%2 == 0`) and logs it to the console. The function is then called with `printEven4to22();`. The terminal shows the output of the command `node printEvenNumbers.js`, displaying the even numbers 4, 6, 8, 10, 12, 14, 16, 18, 20, and 22 on separate lines. The Explorer sidebar on the left lists several other JavaScript files under the 'BASIC OF PROGRAMMING HANDS ON 2' folder.

```
function printEven4to22(){
  for(a=4; a<=22; a++){
    if(a%2 == 0){
      console.log(a);
    }
  }
}

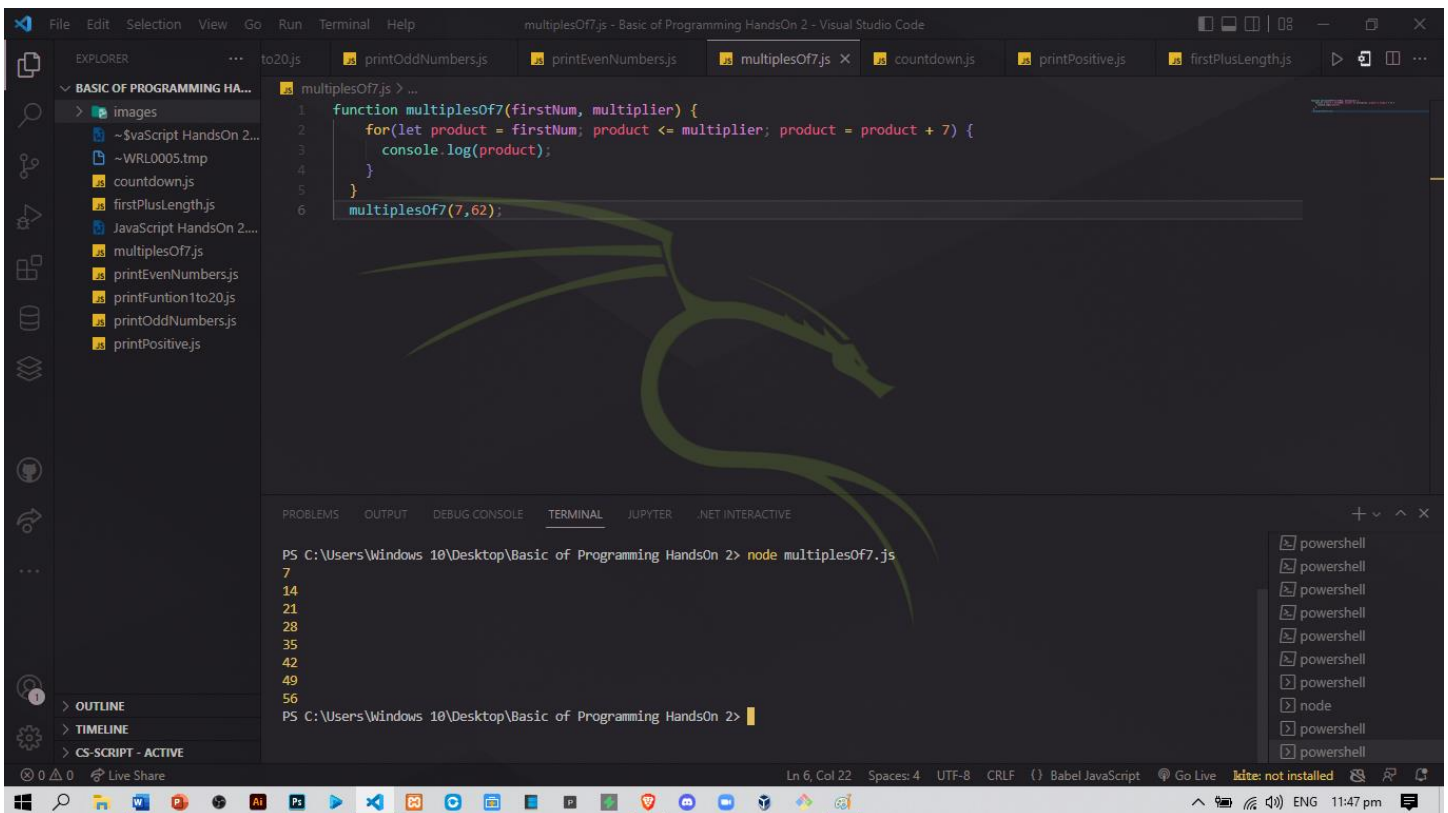
printEven4to22();
```

```
PS C:\Users\Windows 10\Desktop\Basic of Programming HandsOn 2> node printEvenNumbers.js
4
6
8
10
12
14
16
18
20
22
PS C:\Users\Windows 10\Desktop\Basic of Programming HandsOn 2>
```

4. Print/log all the multiples of 7 between the numbers 7 to 62.

Test Cases (0/1)

- multiplesOf7() to log 7 14 21 28 35 42 49 56



The screenshot displays the Visual Studio Code interface. The Explorer panel on the left shows a project named 'BASIC OF PROGRAMMING HANDS ON 2' with various JavaScript files. The main editor window shows the file 'multiplesOf7.js' with the following code:

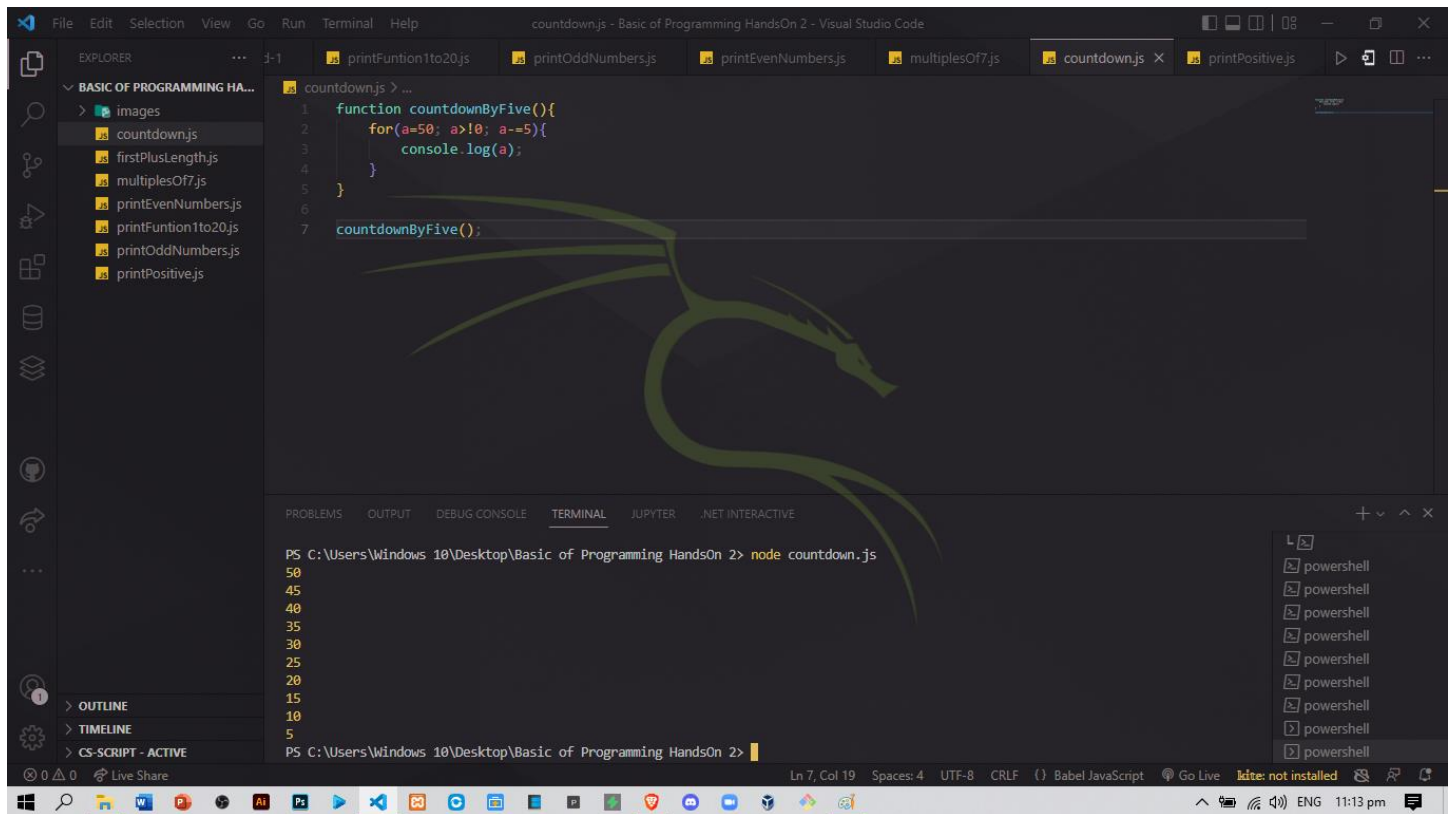
```
1 function multiplesOf7(firstNum, multiplier) {  
2     for(let product = firstNum; product <= multiplier; product = product + 7) {  
3         console.log(product);  
4     }  
5 }  
6 multiplesOf7(7,62);
```

The TERMINAL panel at the bottom shows the command 'node multiplesOf7.js' being executed, resulting in the output: 7, 14, 21, 28, 35, 42, 49, 56. The status bar at the bottom indicates the file is 'Babel JavaScript' and the editor is in 'UTF-8' encoding.

5. Log positive numbers starting at 50, counting down by fives (exclude 0).

Test Cases (0/1)

- countdownByFives() to log 50 45 40 35 30 25 20 15 10 5



The screenshot shows the Visual Studio Code editor with a file named `countdown.js` open. The file contains the following JavaScript code:

```
1 function countdownByFive(){
2   for(a=50; a>!0; a-=5){
3     console.log(a);
4   }
5 }
6
7 countdownByFive();
```

The terminal output shows the execution of the code, logging the numbers 50, 45, 40, 35, 30, 25, 20, 15, 10, and 5. The terminal also shows the command `node countdown.js` being executed.

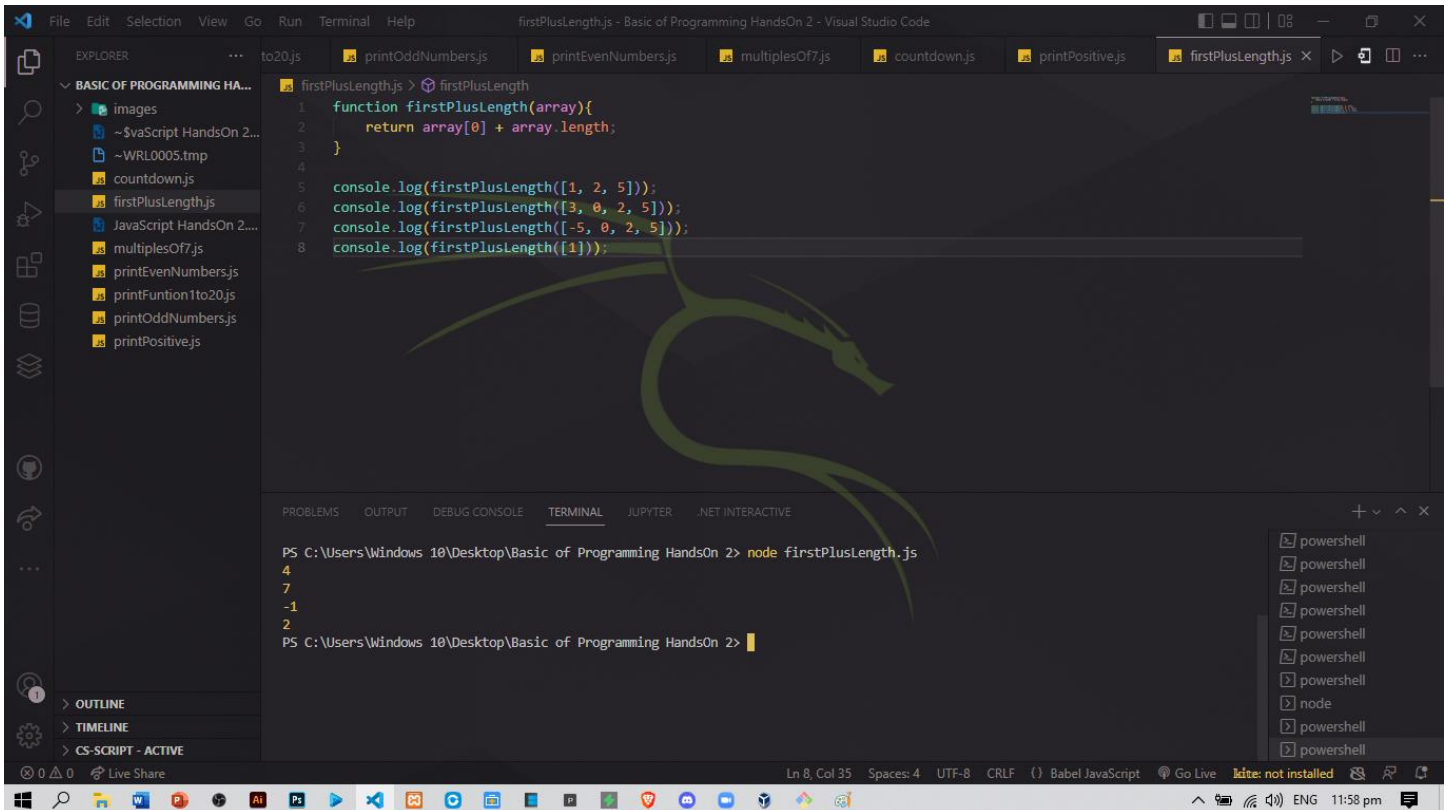
The Explorer sidebar on the left shows the file structure of the project, including `countdown.js` and other files like `printFunction1to20.js`, `printOddNumbers.js`, `printEvenNumbers.js`, `multiplesOf7.js`, and `printPositive.js`.

The bottom status bar indicates the current line and column (Ln 7, Col 19), the encoding (UTF-8), the line ending (CRLF), and the active language (Babel JavaScript).

6. Given an array, print/log the sum of the first value in the array, plus the array's length. Assume that the array is composed of numbers.

Test Cases (0/4)

- `firstPlusLength([1,2,5])` to log 4
- `firstPlusLength([3,0,2,5])` to log 7
- `firstPlusLength([-5,0,2,5])` to log -1
- `firstPlusLength([1])` to log 2



The screenshot shows a Visual Studio Code editor with a file explorer on the left containing various JavaScript files. The main editor window displays the `firstPlusLength.js` file with the following code:

```
1 function firstPlusLength(array){
2   return array[0] + array.length;
3 }
4
5 console.log(firstPlusLength([1, 2, 5]));
6 console.log(firstPlusLength([3, 0, 2, 5]));
7 console.log(firstPlusLength([-5, 0, 2, 5]));
8 console.log(firstPlusLength([1]));
```

Below the code editor, the TERMINAL panel shows the command `node firstPlusLength.js` being executed, resulting in the following output:

```
PS C:\Users\Windows 10\Desktop\Basic of Programming HandsOn 2> node firstPlusLength.js
4
7
-1
2
PS C:\Users\Windows 10\Desktop\Basic of Programming HandsOn 2>
```

The status bar at the bottom indicates the current file is `firstPlusLength.js` at line 8, column 35, with a UTF-8 encoding and CRLF line endings.

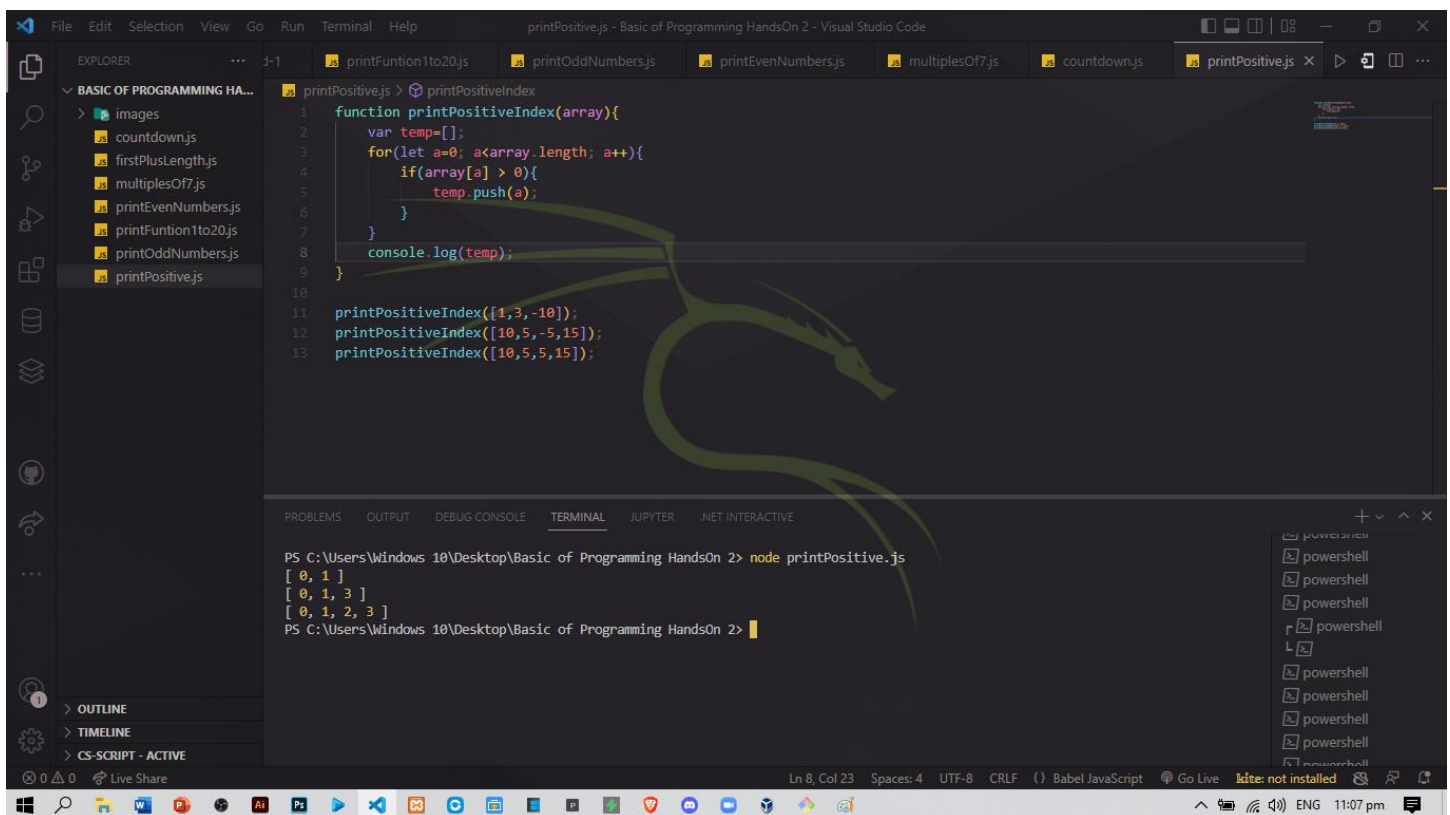
7. Given an array, write a function that prints the index value of its positive values.

For example, `printPositiveIndex([1, 3, -10])`, have it print/log 0, 1 (as the 0th index had a positive value and index 1 also had a positive value).

`printPositiveIndex([10, 5, -5, 15])` should print/log 0, 1, and 3. In other words, it prints the index of each positive number in the array.

Test Cases (0/3)

- `printPositiveIndex([1, 3, -10])` to log 0 1
- `printPositiveIndex([10, 5, -5, 15])` to log 0 1 3
- `printPositiveIndex([10, 5, 5, 15])` to log 0 1 2 3



The screenshot shows a Visual Studio Code editor with a file explorer on the left containing various JavaScript files. The main editor displays the `printPositive.js` file with the following code:

```
1 function printPositiveIndex(array){
2   var temp=[];
3   for(let a=0; a<array.length; a++){
4     if(array[a] > 0){
5       temp.push(a);
6     }
7   }
8   console.log(temp);
9 }
10
11 printPositiveIndex([1,3,-10]);
12 printPositiveIndex([10,5,-5,15]);
13 printPositiveIndex([10,5,5,15]);
```

The terminal at the bottom shows the output of running `node printPositive.js`:

```
PS C:\Users\Windows 10\Desktop\Basic of Programming HandsOn 2> node printPositive.js
[ 0, 1 ]
[ 0, 1, 3 ]
[ 0, 1, 2, 3 ]
PS C:\Users\Windows 10\Desktop\Basic of Programming HandsOn 2>
```

The status bar at the bottom indicates the file is `printPositive.js` at line 8, column 23, using UTF-8 encoding and CRLF line endings. The language mode is set to JavaScript.