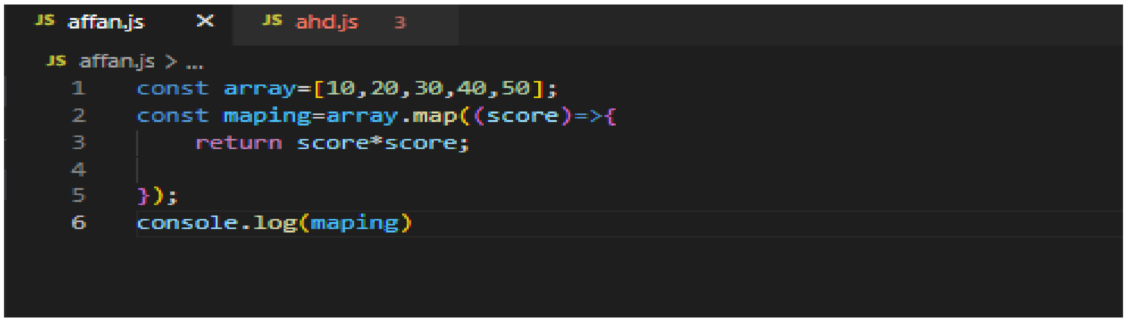
ASSIGNMENT:03
DAY:2,3
JAVA SCRIPT

1. *Map Transformation:*

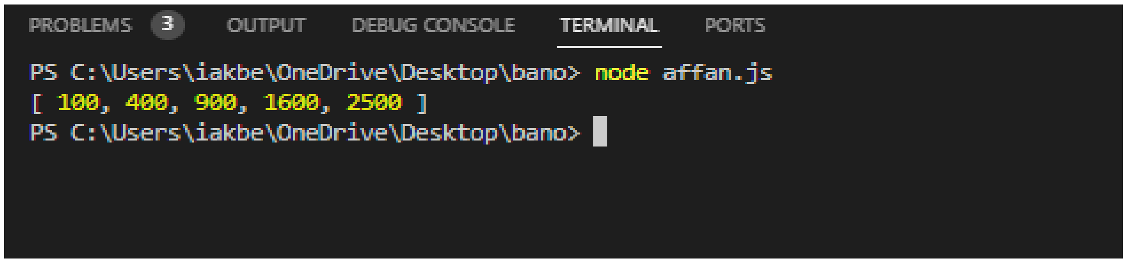
- Q: Given an array of integers, use the map method to square each element and return a new array with the squared values.

SOURCE CODE:



```javascript
const array=[10,20,30,40,50];
const maping=array.map((score)=>{
    return score*score;

});
console.log(maping)
```

OUTPUT:

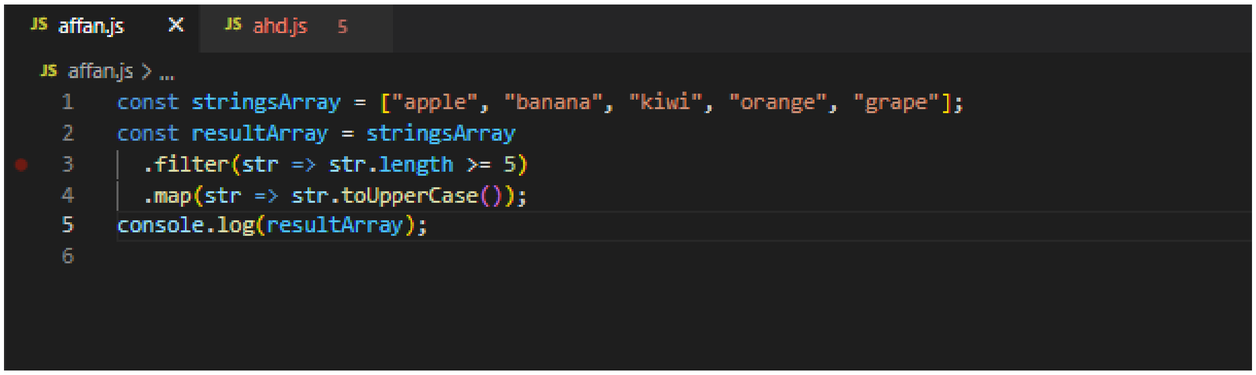


```
PS C:\Users\iakbe\OneDrive\Desktop\bano> node affan.js
[ 100, 400, 900, 1600, 2500 ]
PS C:\Users\iakbe\OneDrive\Desktop\bano>
```

2. *Filter and Map Combination:*

- Q: Take an array of strings, filter out the ones with a length less than 5, and then capitalize the remaining strings using the map method.

SOURCE CODE:

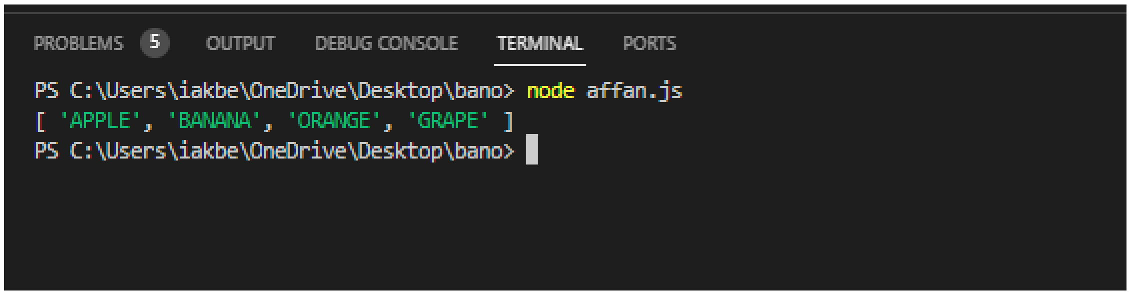


```javascript
const stringsArray = ["apple", "banana", "kiwi", "orange", "grape"];
const resultArray = stringsArray
  .filter(str => str.length >= 5)
  .map(str => str.toUpperCase());
console.log(resultArray);

```

ASSIGNMENT:03

DAY:2,3

OUTPUT:

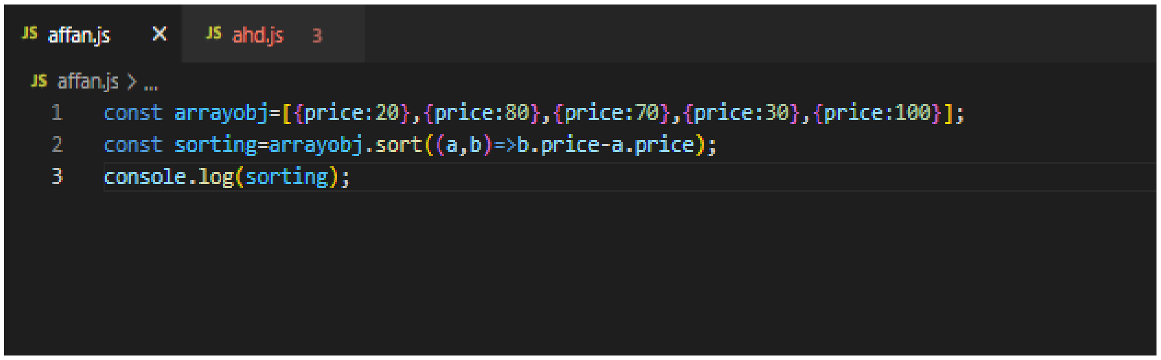

PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS C:\Users\iakbe\OneDrive\Desktop\bano> node affan.js
[ 'APPLE', 'BANANA', 'ORANGE', 'GRAPE' ]
PS C:\Users\iakbe\OneDrive\Desktop\bano>
```

3. *Sorting Objects:*

- Q: Given an array of objects with a 'price' property, use the sort method to arrange them in descending order based on their prices.

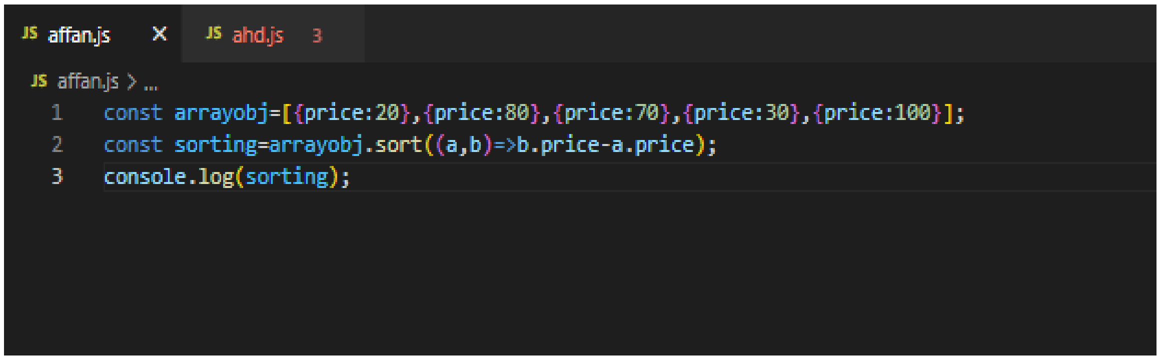
SOURCE CODE:


affan.js ahd.js 3

affan.js > ...

```
const arrayobj=[{price:20},{price:80},{price:70},{price:30},{price:100}];
const sorting=arrayobj.sort((a,b)=>b.price-a.price);
console.log(sorting);
```

OUTPUT:


affan.js ahd.js 3

affan.js > ...

```
const arrayobj=[{price:20},{price:80},{price:70},{price:30},{price:100}];
const sorting=arrayobj.sort((a,b)=>b.price-a.price);
console.log(sorting);
```

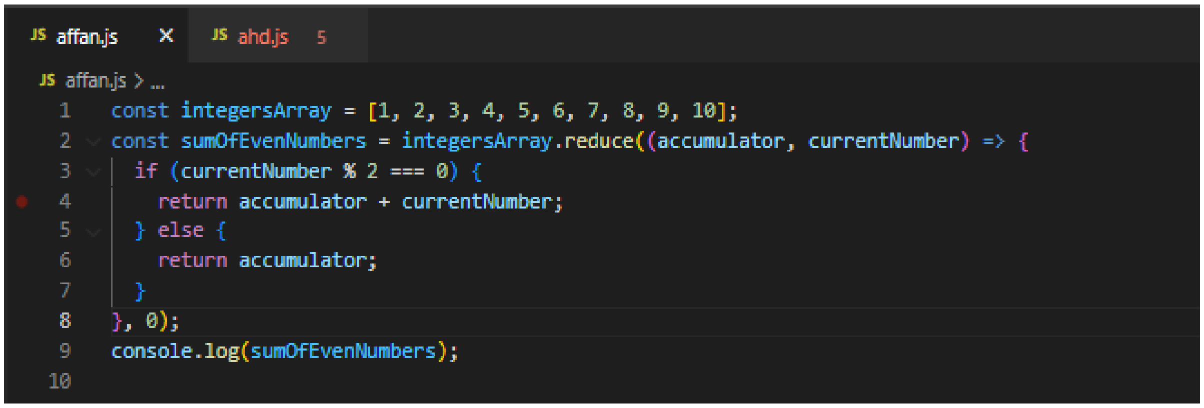
ASSIGNMENT:03

DAY:2,3

4. *Reduce for Aggregation:*

- Q: Use the reduce method to find the total sum of all even numbers in an array of integers.

SOURCE CODE:

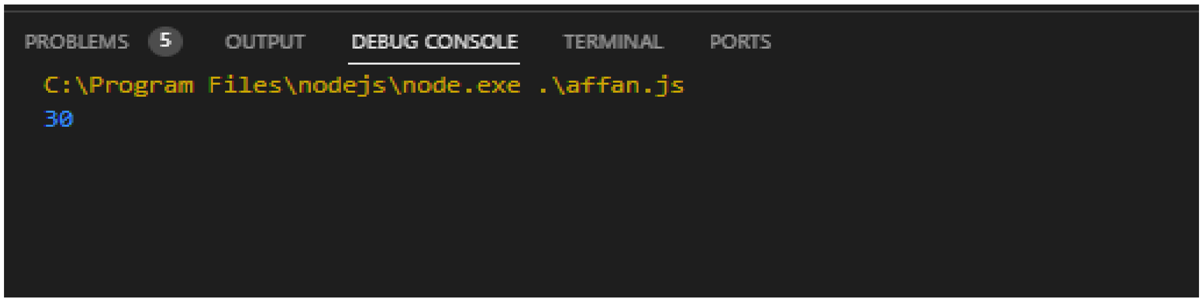


affan.js × ahd.js 5

affan.js > ...

```javascript
const integersArray = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
const sumOfEvenNumbers = integersArray.reduce((accumulator, currentNumber) => {
  if (currentNumber % 2 === 0) {
    return accumulator + currentNumber;
  } else {
    return accumulator;
  }
}, 0);
console.log(sumOfEvenNumbers);
```

OUTPUT:



PROBLEMS 5 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
C:\Program Files\nodejs\node.exe .\affan.js
30
```

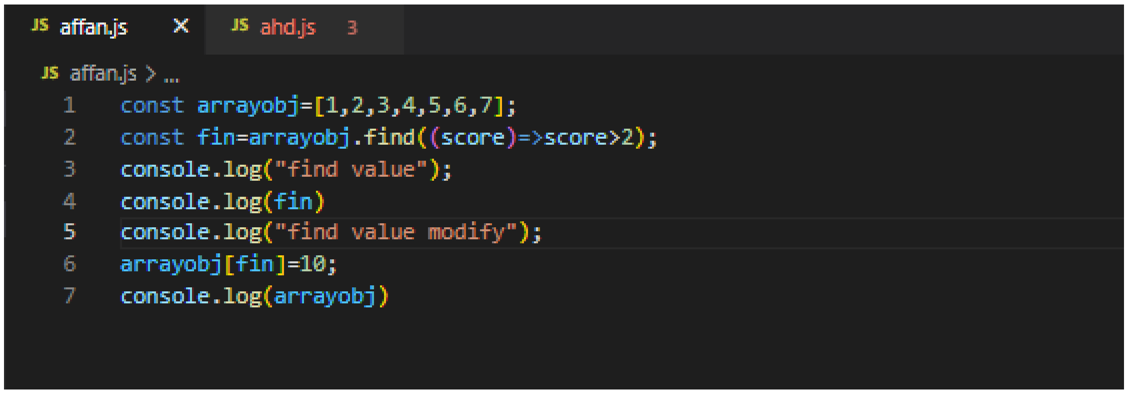
5. *Find and Modify:*

- Q: Given an array of objects with 'id' properties, use the find method to locate an object with a specific 'id' and update its 'status' property to 'completed'.

SOURCE CODE:

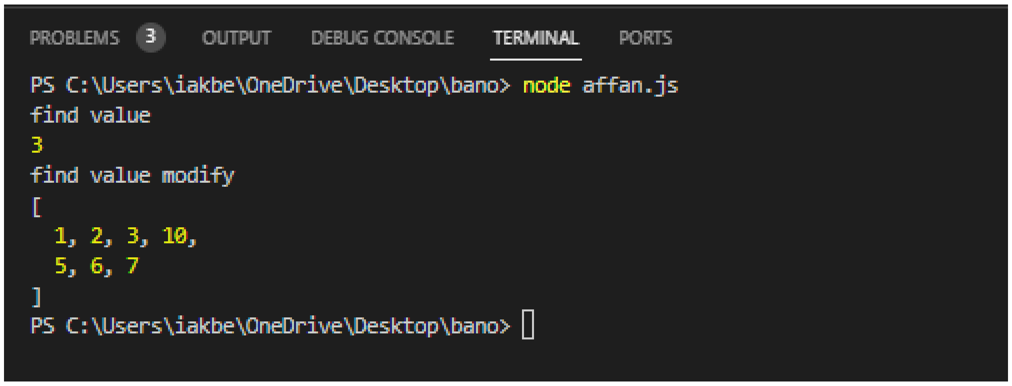
ASSIGNMENT:03

DAY:2,3



```
const arrayobj=[1,2,3,4,5,6,7];
const fin=arrayobj.find((score)=>score>2);
console.log("find value");
console.log(fin)
console.log("find value modify");
arrayobj[fin]=10;
console.log(arrayobj)
```

OUTPUT:

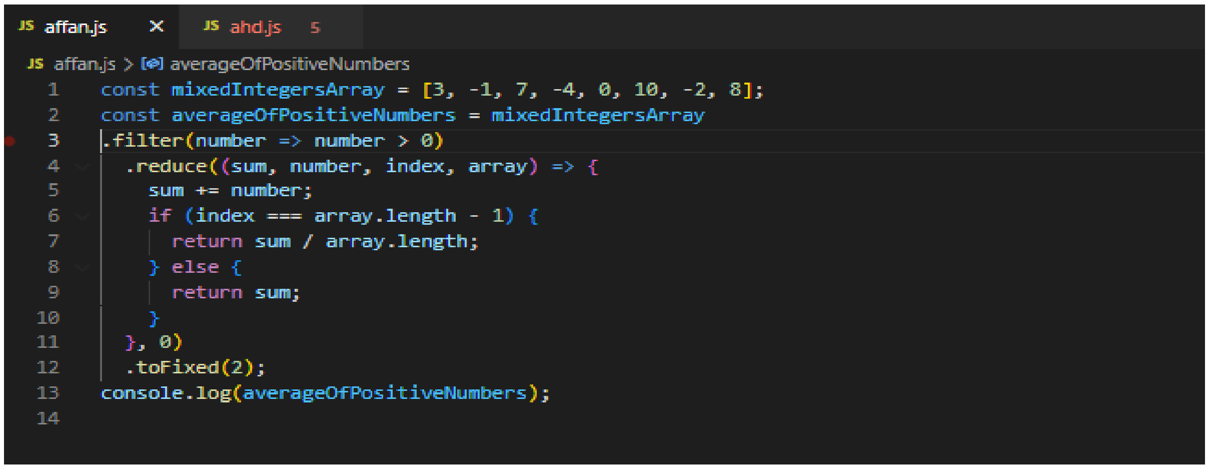


```
PS C:\Users\iakbe\OneDrive\Desktop\bano> node affan.js
find value
3
find value modify
[
  1, 2, 3, 10,
  5, 6, 7
]
PS C:\Users\iakbe\OneDrive\Desktop\bano>
```

6. *Chaining Methods:*

- Q: Create a chain of array methods to find the average of all positive numbers in an array of mixed integers and return the result rounded to two decimal places.

SOURCE CODE:

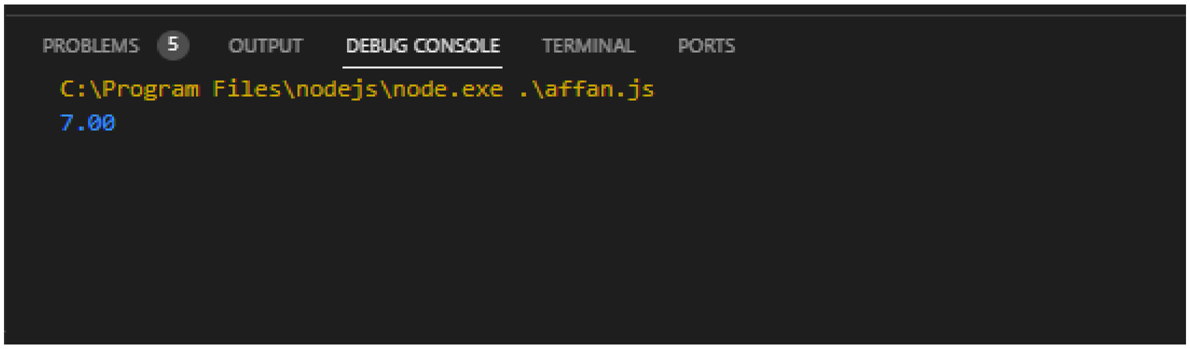


```
const mixedIntegersArray = [3, -1, 7, -4, 0, 10, -2, 8];
const averageOfPositiveNumbers = mixedIntegersArray
.filter(number => number > 0)
  .reduce((sum, number, index, array) => {
    sum += number;
    if (index === array.length - 1) {
      return sum / array.length;
    } else {
      return sum;
    }
  }, 0)
  .toFixed(2);
console.log(averageOfPositiveNumbers);
```

OUTPUT:

ASSIGNMENT:03

DAY:2,3

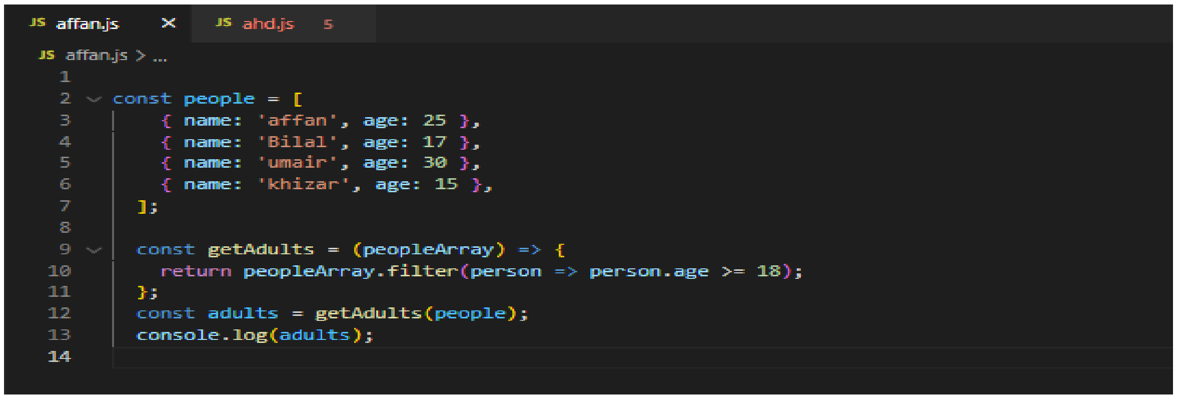


```
C:\Program Files\nodejs\node.exe .\affan.js
7.00
```

7. *Conditional Filtering:*

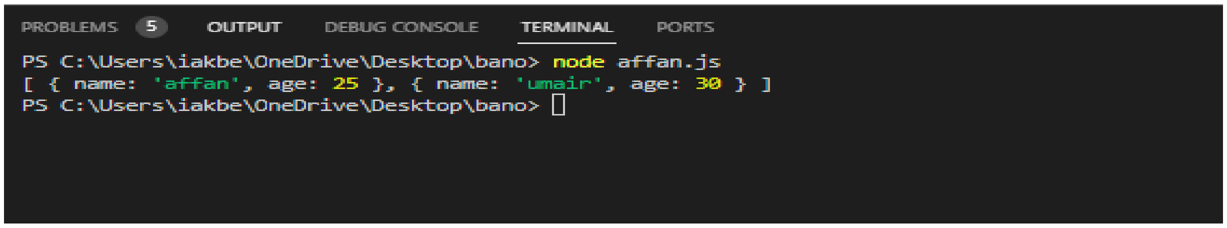
- Q: Implement a function that takes an array of objects with 'age' properties and returns an array of those who are adults (age 18 and above) using the filter method.

SOURCE CODE:



```

const people = [
    { name: 'affan', age: 25 },
    { name: 'Bilal', age: 17 },
    { name: 'umair', age: 30 },
    { name: 'khizar', age: 15 },
  ];

  const getAdults = (peopleArray) => {
    return peopleArray.filter(person => person.age >= 18);
  };
  const adults = getAdults(people);
  console.log(adults);

```

OUTPUT:



```
PS C:\Users\iakbe\OneDrive\Desktop\bano> node affan.js
[ { name: 'affan', age: 25 }, { name: 'umair', age: 30 } ]
PS C:\Users\iakbe\OneDrive\Desktop\bano> 
```

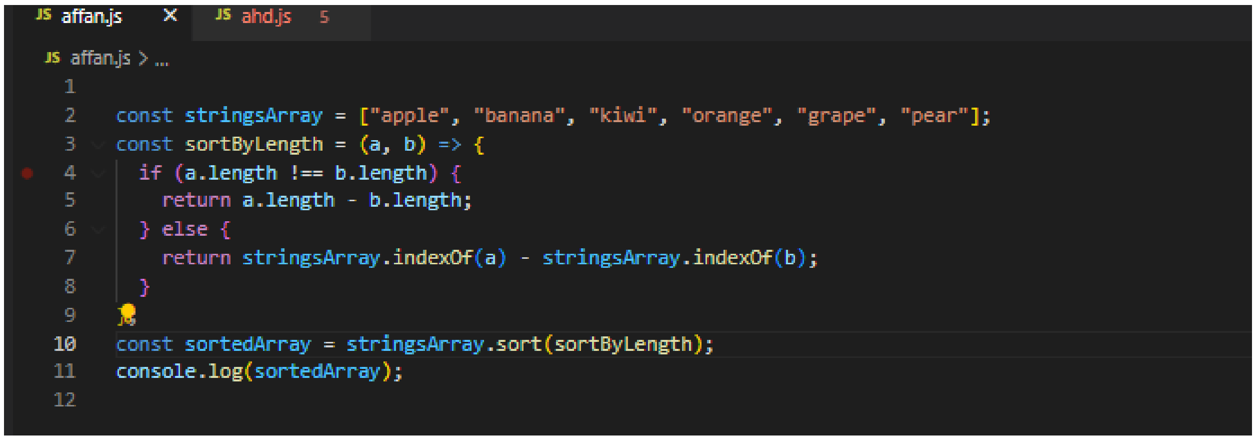
8. *Advanced Sorting:*

- Q: Sort an array of strings based on their lengths in ascending order. If two strings have the same length, maintain their relative order in the sorted array.

SOURCE CODE:

ASSIGNMENT:03

DAY:2,3

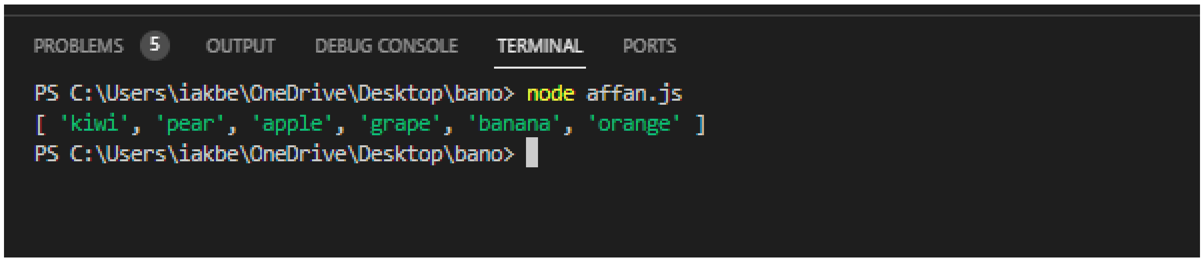


affan.js ✕   ahd.js 5

affan.js > ...

```javascript

const stringsArray = ["apple", "banana", "kiwi", "orange", "grape", "pear"];
const sortByLength = (a, b) => {
  if (a.length !== b.length) {
    return a.length - b.length;
  } else {
    return stringsArray.indexOf(a) - stringsArray.indexOf(b);
  }
};
const sortedArray = stringsArray.sort(sortByLength);
console.log(sortedArray);

```

OUTPUT:



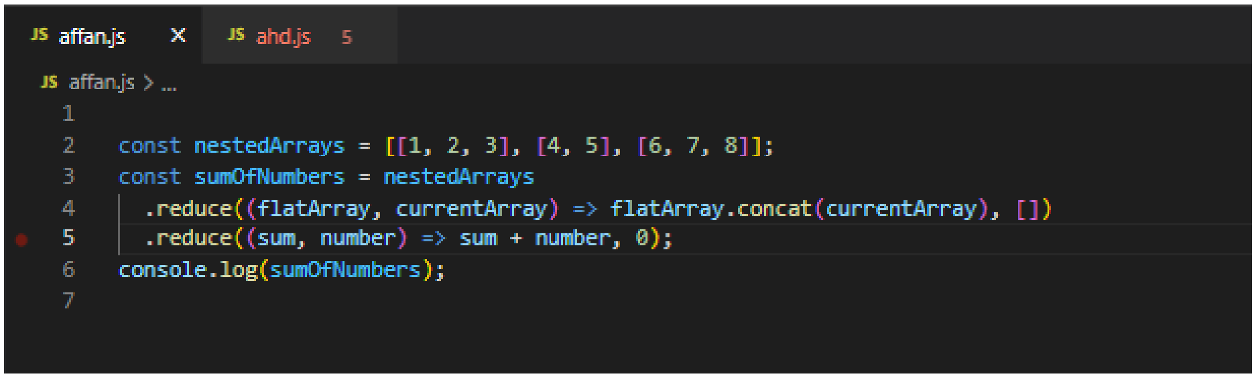
PROBLEMS 5   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

```
PS C:\Users\iakbe\OneDrive\Desktop\bano> node affan.js
[ 'kiwi', 'pear', 'apple', 'grape', 'banana', 'orange' ]
PS C:\Users\iakbe\OneDrive\Desktop\bano>
```

9. *Nested Array Operations:*

- Q: Given an array of arrays containing numbers, use a combination of array methods to flatten the structure and then calculate the sum of all the numbers.

SOURCE CODE:



affan.js ✕   ahd.js 5

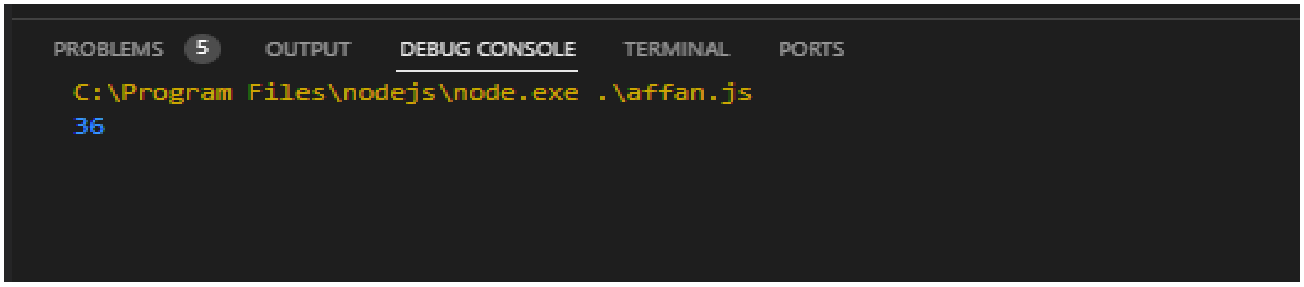
affan.js > ...

```javascript

const nestedArrays = [[1, 2, 3], [4, 5], [6, 7, 8]];
const sumOfNumbers = nestedArrays
  .reduce((flatArray, currentArray) => flatArray.concat(currentArray), [])
  .reduce((sum, number) => sum + number, 0);
console.log(sumOfNumbers);

```

OUTPUT:

ASSIGNMENT:03

DAY:2,3

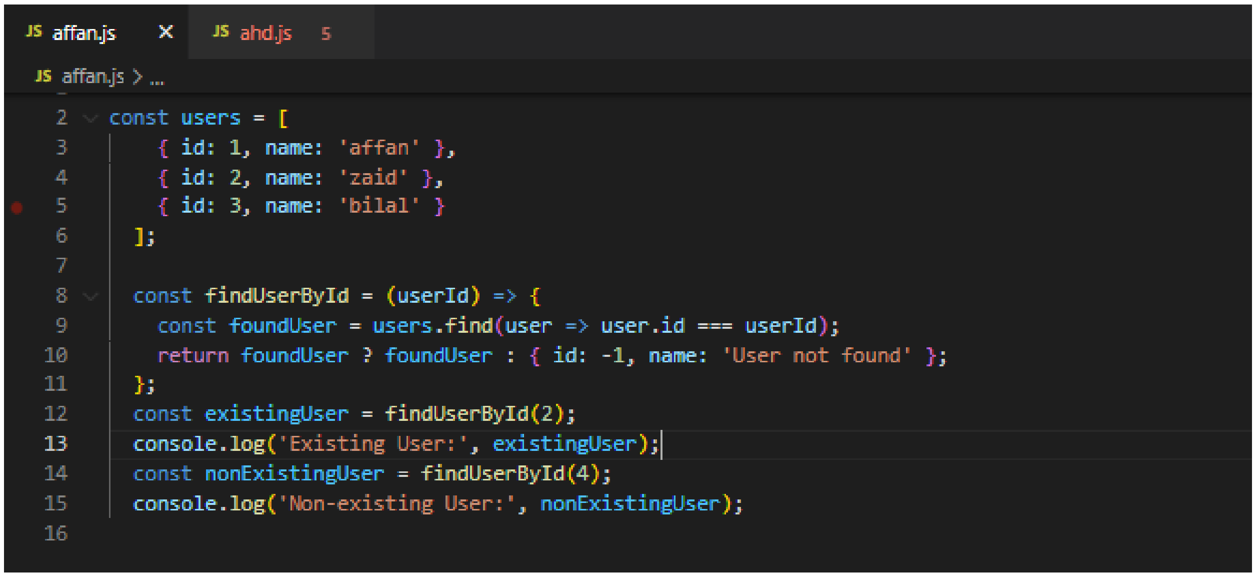


```
C:\Program Files\nodejs\node.exe .\affan.js
36
```

10. *Error Handling with Find:*

- Q: Modify the find method to handle the scenario where the desired element is not found, returning a custom default object instead.

SOURCE CODE:

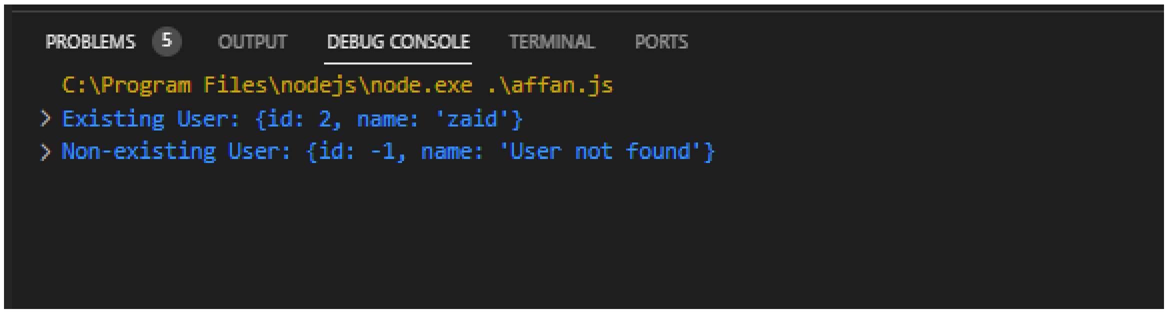


```javascript
const users = [
    { id: 1, name: 'affan' },
    { id: 2, name: 'zaid' },
    { id: 3, name: 'bilal' }
  ];

  const findUserById = (userId) => {
    const foundUser = users.find(user => user.id === userId);
    return foundUser ? foundUser : { id: -1, name: 'User not found' };
  };
  const existingUser = findUserById(2);
  console.log('Existing User:', existingUser);
  const nonExistingUser = findUserById(4);
  console.log('Non-existing User:', nonExistingUser);
```

OUTPUT:

ASSIGNMENT:03

DAY:2,3



```
  C:\Program Files\nodejs\node.exe .\affan.js
> Existing User: {id: 2, name: 'zaid'}
> Non-existing User: {id: -1, name: 'User not found'}
```