ASSIGNMENT:02
DAY:03
JAVA SCRIPT

11) Write a for...in loop that iterates over the properties of an object and logs each property name and value.

SOURCE CODE:

affan.js

```
let obj={
  name:"affan",
  occ:"engineer",
};
for (let value in obj ){
  console.log(obj[value]);
}
```

OUTPUT:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
C:\Program Files\nodejs\node.exe .\affan.js
affan
engineer
```

12) 12. Explain the use of the break and continue statements within loops. Provide scenarios where each might be used.

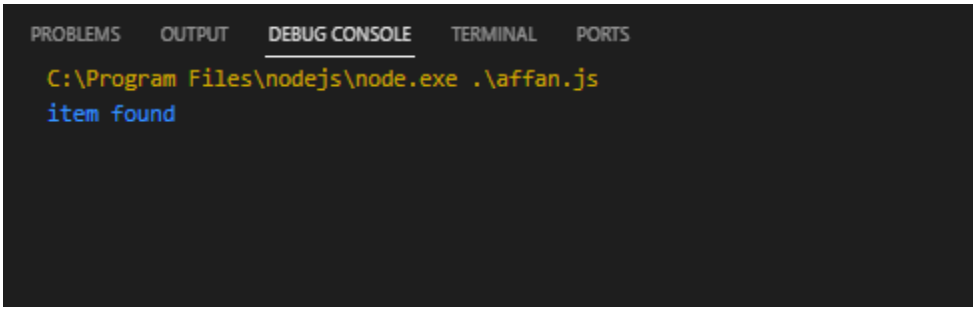
SOURCE CODE: (break statement)

affan.js

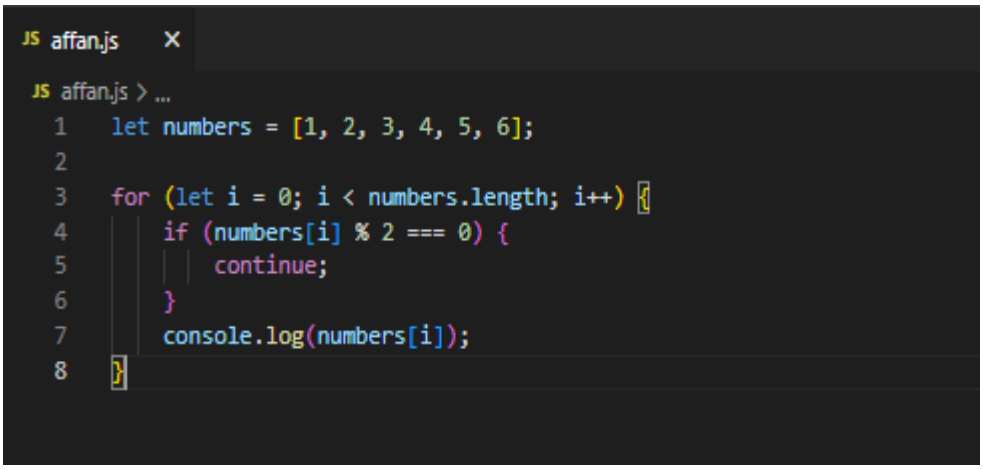
```
let targetItem = 42;
let array = [10, 23, 42, 15, 30];

for (let i = 0; i < array.length; i++) {
    if (array[i] === targetItem) {
        console.log("item found");
        break;
    }
}
```

ASSIGNMENT:02
DAY:03

OUTPUT:



SOURCE CODE:(Continue Statement)



OUTPUT:

ASSIGNMENT:02
DAY:03

13) Write a function calculateTax that calculates and returns the tax amount based on a given income. Use a ternary operator to determine the tax rate.

SOURCE CODE:

affan.js

```
function calculateTax(income) {
    const lowRate = 0.1;
    const highRate = 0.2;
    const taxRate = income > 50000 ? highRate : lowRate;
    const taxAmount = income * taxRate;
    return taxAmount;
}
const income1 = 40000;
const tax1 = calculateTax(income1);
console.log("the first income:"+tax1);

const income2 = 60000;
const tax2 = calculateTax(income2);
console.log("the second income:"+tax2);
```

OUTPUT:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
C:\Program Files\nodejs\node.exe .\affan.js
the first income:4000
the second income:12000
```

14) Create an object car with properties make, model, and a method startEngine that logs a message. Instantiate the object and call the method.

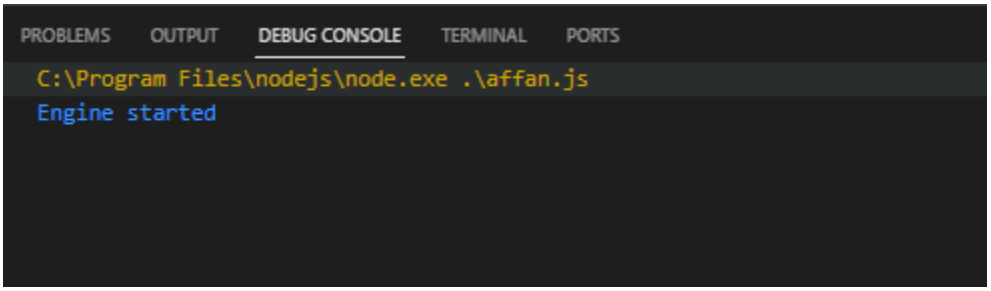
SOURCE CODE:

affan.js

```
const car = {
    make: "Toyota",
    model: "2020",
    startEngine: function() {
        console.log("Engine started");
    }
};

car.startEngine();
```

ASSIGNMENT:02
DAY:03

OUTPUT:



15) Explain the differences between regular functions and arrow functions in terms of scope, this binding, and their use as methods.

SOURCE CODE:

affan.js

```
// regular function
function sum0ftw0num(a,b){
    sum=a+b;
    return sum;
}
const cal= sum0ftw0num(2,3);
console.log("regular function")
console.log(cal)
// arrow function
const sumnumbers=(a,b)=>{
    result=a+b;
    return result;
}
const sum2 = sumnumbers(2,3);
console.log("Arrow function")
console.log(sum2)
```

OUTPUT:

ASSIGNMENT:02
DAY:03