

**Q. Create a module that exports a function and import it into another file.**

The screenshot shows the OneCompiler web interface. The editor has two files: `index.js` and `script1.js`. `script1.js` contains a module that exports functions for calculating the area and perimeter of a rectangle. `index.js` imports this module and calls the functions with specific values. The output shows the results of these calculations.

```
1 const lib = require('./script1.js');
2 let length = 10;
3 let breadth = 5;
4
5 lib.area(length, breadth);
6 lib.perimeter(length, breadth);
7
```

Output:

```
Area of the rectangle is 50 square unit
Perimeter of the rectangle is 30 unit
```

**Q. Convert a JavaScript object to JSON and back using `JSON.stringify()` and `JSON.parse()`.**

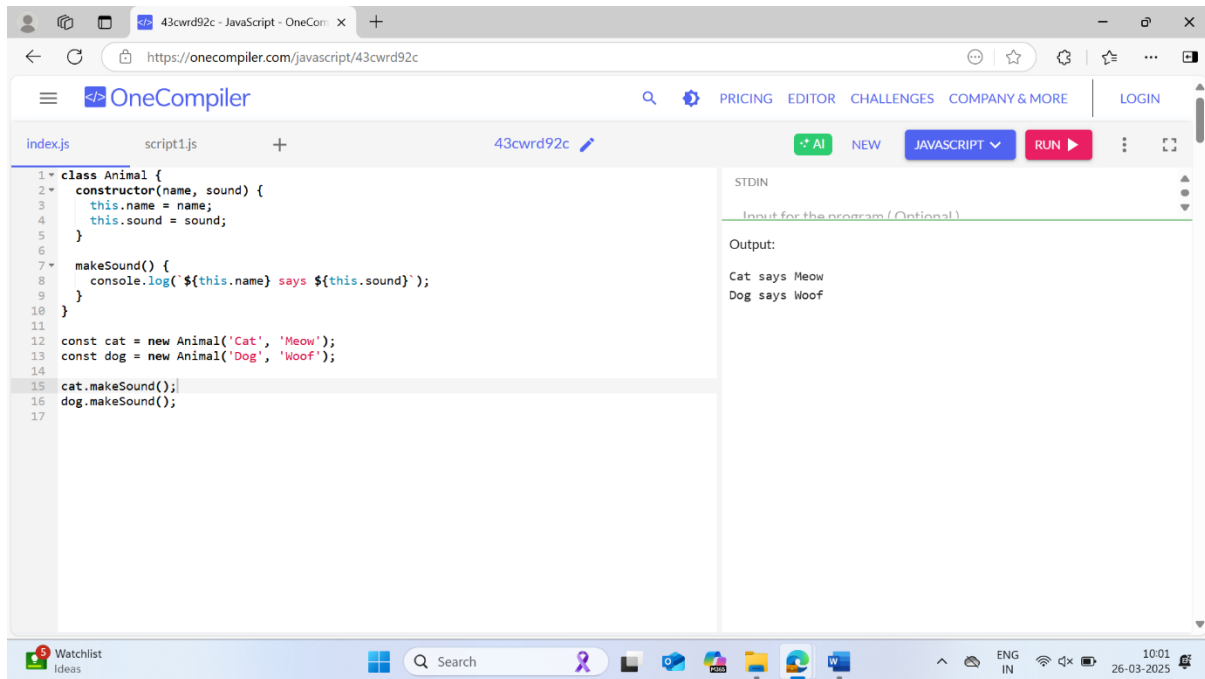
The screenshot shows the OneCompiler web interface. The editor has two files: `index.js` and `script1.js`. `script1.js` exports an object with an `add` function and the value of `pi`. `index.js` imports this object, logs the `add` function's result, and logs the `pi` value. The output shows the results of these operations.

```
1 const { add, pi } = require('./script1.js');
2
3 console.log(add(2, 3));
4 console.log(pi);
5
```

Output:

```
5
3.14159
```

**Q. Define a class Animal with properties and a method, and create instances.**



The screenshot shows the OneCompiler web interface. The code editor on the left contains the following JavaScript code:

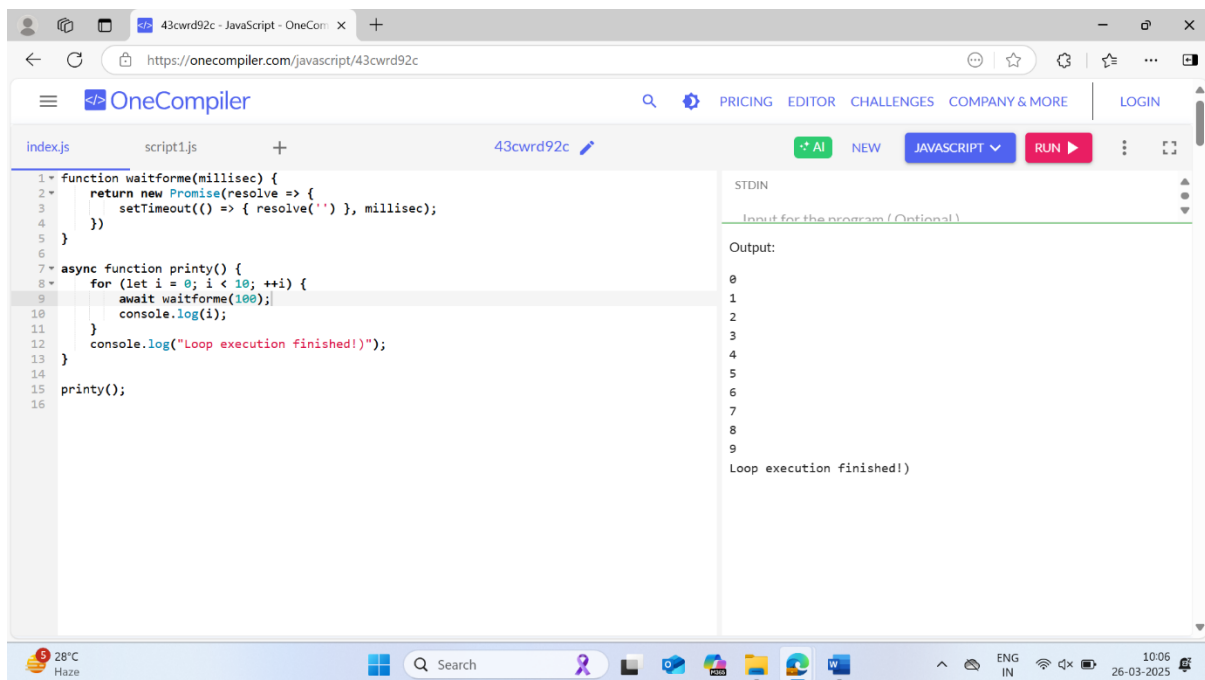
```
1 class Animal {
2   constructor(name, sound) {
3     this.name = name;
4     this.sound = sound;
5   }
6
7   makeSound() {
8     console.log(`${this.name} says ${this.sound}`);
9   }
10 }
11
12 const cat = new Animal('Cat', 'Meow');
13 const dog = new Animal('Dog', 'Woof');
14
15 cat.makeSound();
16 dog.makeSound();
17
```

The output panel on the right shows the following output:

```
STDIN
Input for the program (Optional)

Output:
Cat says Meow
Dog says Woof
```

**Q. Write a promise that resolves after a delay and convert it into an async function using async/await.**



The screenshot shows the OneCompiler web interface. The code editor on the left contains the following JavaScript code:

```
1 function waitforme(millsec) {
2   return new Promise(resolve => {
3     setTimeout(() => { resolve('') }, millsec);
4   })
5 }
6
7 async function printy() {
8   for (let i = 0; i < 10; ++i) {
9     await waitforme(100);
10    console.log(i);
11  }
12  console.log("Loop execution finished!");
13 }
14
15 printy();
16
```

The output panel on the right shows the following output:

```
STDIN
Input for the program (Optional)

Output:
0
1
2
3
4
5
6
7
8
9
Loop execution finished!
```