Advanced JavaScript Part 2

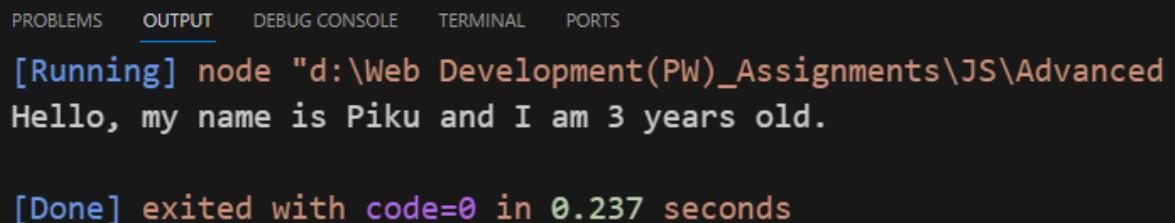[Assignment Questions with Answers]

Problem 1: Create an object constructor Person that takes name and age as parameters and initializes them. Also, add a method sayHello to greet the person.

Ans.

```
class Person
{
  constructor(name, age)
  {
    this.name = name;
    this.age = age;
  }
  sayHello()
  {
    console.log("Hello, my name is " + this.name + " and I am " + this.age + " years old.");
  };
}
const insOfPerson = new Person("Piku", 3);
insOfPerson.sayHello();
```

Output:


```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS

[Running] node "d:\Web Development(PW)_Assignments\JS\Advanced
Hello, my name is Piku and I am 3 years old.

[Done] exited with code=0 in 0.237 seconds
```

Problem 2: Create a constructor Employee that inherits from the Person constructor of problem 1. Add an additional property designation and a method getDetails to display the employee details.

Ans.

```
class Person
{
  constructor(name, age)
  {
   //initializes properties
   this.name = name;
   this.age = age;
  }
}
class Employee extends Person
{
  constructor(name, age,designation)
  {
   //initializes properties
   super(name,age);
   this.designation = designation;
  }
  getDetails()
  {
   console.log("Hello, my name is " + this.name + " and I am " + this.age + " years old " +
"and designation is " + this.designation);
  }
}
const insOfEmployee = new Employee("Piku",28,"Manager");
insOfEmployee.getDetails();
```

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

[Running] node "d:\Web Development(PW)_Assignments\JS\Advanced JavaScrip
Hello, my name is Piku and I am 28 years old and designation is Manager

[Done] exited with code=0 in 1.168 seconds
```

Problem 3: Create an object Calculator with methods add, subtract, multiply, and divide. Demonstrate the usage of this within these methods such that method chaining of add, subtract, multiply and divide is possible.
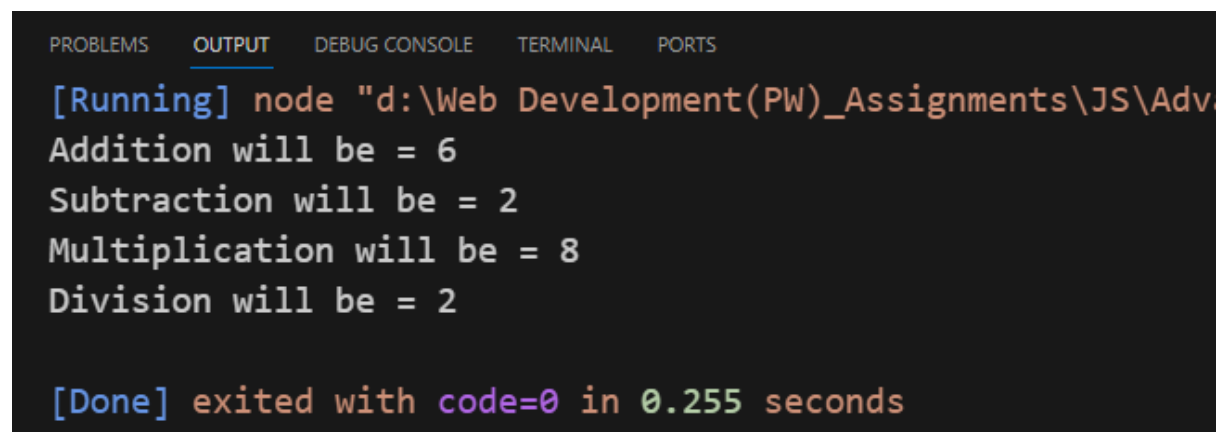
Ans.

```
class Calculator

{

  constructor(num1,num2)

  {

    this.num1 = num1;

    this.num2 = num2;

  }

  add()

  {

    return this.num1 + this.num2 ;

  }

  subtract()

  {

    return this.num1 - this.num2 ;

  }

  multiply()

  {

    return this.num1 * this.num2 ;

  }
```

```
    divide()

    {

        return this.num1 / this.num2 ;

    }

}
```

const insOfCalculator = new Calculator(4,2);

console.log("Addition will be =",insOfCalculator.add());

console.log("Subtraction will be =",insOfCalculator.subtract());

console.log("Multiplication will be =",insOfCalculator.multiply());

console.log("Division will be =",insOfCalculator.divide());

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

[Running] node "d:\Web Development(PW)_Assignments\JS\Adv
Addition will be = 6
Subtraction will be = 2
Multiplication will be = 8
Division will be = 2


[Done] exited with code=0 in 0.255 seconds
```

Problem 4: Define a base class Shape with a method draw. Create two subclasses Circle and Rectangle that override the draw method. Demonstrate polymorphism using instances of these classes.

Ans.

```
class Shape

{

    constructor(name)

    {

        this.name = name;

    }
```
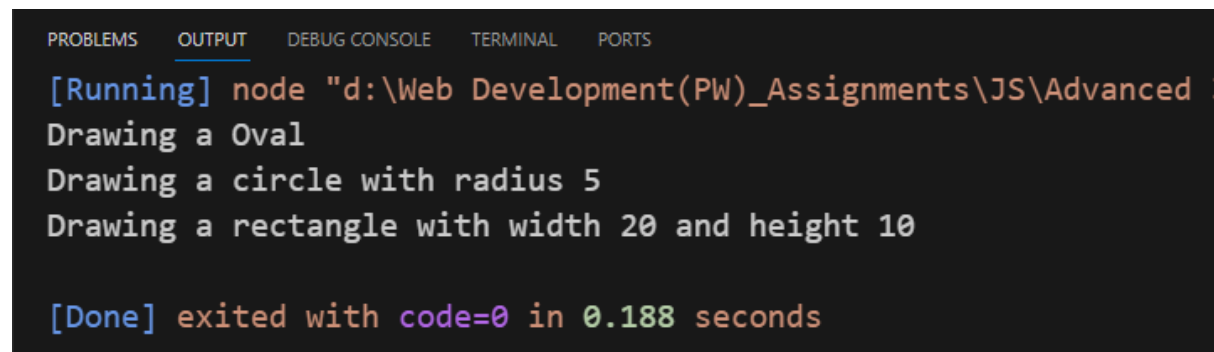
```javascript
    draw()
    {
      console.log(`Drawing a ${this.name}`);
    }
}
class Circle extends Shape
{
    constructor(radius)
    {
      super('Circle');
      this.radius = radius;
    }
    draw()
    {
      console.log(`Drawing a circle with radius ${this.radius}`);
    }
}
class Rectangle extends Shape
{
    constructor(height,width)
    {
      super('Rectangle');
      this.height = height;
      this.width = width;
    }
    draw()
    {
      console.log(`Drawing a rectangle with width ${this.width} and height ${this.height}`);
    }
```

```
}
```

// Demonstrate polymorphism using instances of these classes

const shape = new Shape("Oval")

const circle = new Circle(5);

const rectangle = new Rectangle(10, 20);

shape.draw();

circle.draw();

rectangle.draw();

Output:

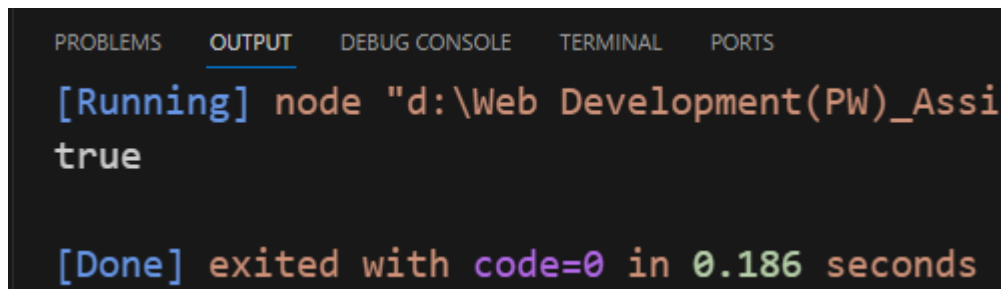Problem 5: Create a simple polyfill for the Array.includes method by the name of customIncludes

Ans.

Array.prototype.customIncludes = null;

if (!Array.prototype.customIncludes) {

   Array.prototype.customIncludes = function(element)

  {

     for (let i = 0; i < this.length; i++)

    {

      if (this[i] === element)

      {

```
            return true;
        }
    }
    return false;
  };
}
const arr = [2,8,5,9];
console.log(arr.customIncludes(5));
```

Output:

THE END