```
module.exports = function delay(time, object, method, args) {
    return new Promise((resolve, reject) => {
        setTimeout(function(){
            if(method in object){
                resolve(object[method].apply(this, args));
            }else{
                reject(`Object has no method: ${method}`);
            }
        }, time);
    });
};
```

```
# JPMC Interview

* GitHub: https://github.com/Toosick/jpmc-interview
```

```
class Calculator {
    add(lhs, rhs) {
        return lhs + rhs;
    }

    subtract(lhs, rhs) {
        return lhs - rhs;
    }

    multiply(lhs, rhs) {
        return lhs * rhs;
    }

    divide(lhs , rhs) {
        if (rhs === 0 || isNaN(rhs)) {
            return NaN;
        }else {
            return lhs / rhs;
        }
    }
}

module.exports = Calculator;
```

```
const withExponents = function withExponents() {
    Object.assign(this, {
        pow(base, exponent) {
            return Math.pow(base, exponent);
        },

        multiplyExp(lhs, rhs) {
            return this.multiply(this.pow(...lhs), this.pow(...rhs));
        },

        divideExp(lhs, rhs) {
            return this.divide(this.pow(...lhs), this.pow(...rhs));
        }
    });
};

module.exports = withExponents;
```

```
# Product List Components
* search_results
* category_list
    * category
* department_sorter
    * Data: clothing, computers, ...
    * Actions: open, close
    * Configuration
        * choose_label
        * sort_label
        * down_arrow_url
        * departments_label
* past_order
* advertisement
* related_search
* product_list
    * product_list_item
        * product_buy_info
        * product_reviews

# Product Details Components
* header_navigation
* advertisement
* header_bar
    * shop_by
    * search_panel
        * search_by
    * account_dropdown
    * cart_dropdown
    * wish_list_dropdown
* product_breadcrumbs
    * product_breakcrumb
* social_share
* product_add_to
    * product_add_to_cart
    * product_single_sign_on
    * product_add_to_wish_list
* product_other_sellers_product_list
    * seller_product_list_item
* product_image_viewer
    * Data: img_1_url, img_2_url, img_1_thumbnail_url, img_2_thumbnail_url
    * Actions: mouse_in (zoom), change_img, mouse_out(zoom), mouse_move (zoom)
    * Configuration
        * img_dimensions
        * roll_over_label
        * zoom_dimensions
        * zoom_position
        * zoom_window_percentage
        * thumbnail_dimensions
* product_info
* product_buy_info
    * product_shipping
* product_types
* product_bullet_point_list
    * product_bullet_point
* product_other_sellers_info
* similar_product_deals

# API Design
Assuming that driving directions, services, and hours is handled with the Bing Javascript Maps API
### Resources
* Search
```

    * GET
        * Input: query, page, count
        * Output
            * Date Type: JSON
            ```
                {
                  "meta": {
                    "page": 1,
                    "total": 199
                  },
                  "banks": [
                    {
                      "bank_id": "1",
                      "address": "770 Polaris Pkwy",
                      "city": "Lewis Center",
                      "state": "OH",
                      "postal": "43035",
                      "center": "Speedway",
                      "phone": "614-217-6284",
                      "hours": {
                        "Monday": "9-5 PM",
                        "Tuesday": "9-5 PM",
                        "Wednesday": "9-5 PM",
                        "Thursday": "9-5 PM",
                        "Friday": "9-5 PM",
                        "Saturday": "12-5 PM",
                        "Sunday": "2-5 PM"
                      },
                      "services": ["atm", "personal_banking", "business_banking", "credit_cards", "loans"],
                      "latitude": "40.151197",
                      "longitude": "-82.997971"
                    }
                  ]
                }
* Bank
    * Definition
        * bank_id
        * address
        * city
        * state
        * postal
        * center
        * phone
        * hours
        * services
        * latitude
        * longitude
            ```

```
const Calculator = require('./calculator');

class ScientificCalculator extends Calculator {
    sin(x) {
        return Math.sin(x)
    }

    cos(x) {
        return Math.cos(x)
    }

    tan(x) {
        return Math.tan(x)
    }

    log(x) {
        return Math.log(x)
    }
}

module.exports = ScientificCalculator;
```

```
const chai = require('chai');
const chaiAsPromised = require('chai-as-promised');
chai.use(chaiAsPromised);
const expect = chai.expect;

const delay = require('../delay');
const Calculator = require('../calculator');

describe( "delay", function(){
    var calculator = new Calculator();
    it( "returns a promise", function(){
        var willAdd = delay( 100, calculator, 'add', [ 1, 1 ] );
        expect( willAdd ).to.be.instanceOf( Promise );
        return expect( willAdd ).to.be.fulfilled;
    } );
    it( "delays execution add", function(){
        return expect( delay( 1000, calculator, 'add', [ 10, 5 ] ) ).to.eventually.equal( 15 );
    } );
    it( "delays execution sub", function(){
        return expect( delay( 500, calculator, 'subtract', [ 9, 5 ] ) ).to.eventually.equal( 4 );
    } );
    it( "cannot execute functions that do not exist", function(){
        return expect( delay( 1000, calculator, 'sqrt', [ 2, 2 ] ) ).to.be.rejected;
    } );
} );
```

File - C:\Users\PrestonSSD2\PhpstormProjects\JPMCTestOutput\test\calculator.js

```
const chai = require('chai');
const expect = chai.expect;

const Calculator = require('../calculator');

describe( "Calculator", function(){
    var calculator;
    beforeEach( function(){
        calculator = new Calculator();
    } );
    it( "adds 1 and 2", function(){
        expect( calculator.add( 1, 2 ) ).to.equal( 3 );
    } );
    it( "subtracts 2 from 9", function(){
        expect( calculator.subtract( 9, 2 ) ).to.equal( 7 );
    } );
    it( "multiplies 4 and 3", function(){
        expect( calculator.multiply( 4, 3 ) ).to.equal( 12 );
    } );
    it( "divides 10 by 2", function(){
        expect( calculator.divide( 10, 2 ) ).to.equal( 5 );
    } );
    it( "does not divide by 0", function(){
        expect( calculator.divide( 5, 0 ) ).to.be.NaN;
    } );
} );
```

```
const chai = require('chai');
const expect = chai.expect;

const withExponents = require('../with-exponents');
const Calculator = require('../calculator');

describe( "withExponents", function(){
    var calculator;
    beforeEach( function(){
        calculator = new Calculator();
        withExponents.call( calculator );
    } );
    it( "returns 2^3", function(){
        expect( calculator.pow( 2, 3 ) ).to.equal( 8 );
    } );
    it( "multiplies 2^3 and 2^4", function(){
        expect( calculator.multiplyExp( [ 2, 3 ], [ 2, 4 ] ) ).to.equal( 128 );
    } );
    it( "divides 2^3 by 2^5", function(){
        expect( calculator.divideExp( [ 2, 3 ], [ 2, 5 ] ) ).to.equal( 0.25 );
    } );
} );
```

```
const chai = require('chai');
const expect = chai.expect;

const ScientificCalculator = require('../scientific-calculator');
const Calculator = require('../calculator');

describe( "ScientificCalculator", function(){
    var calculator;
    beforeEach( function(){
        calculator = new ScientificCalculator();
    } );
    it( "extends Calculator", function(){
        expect( calculator ).to.be.instanceOf( Calculator );
        expect( calculator ).to.be.instanceOf( ScientificCalculator );
    } );
    it( "returns the sine of PI / 2", function(){
        expect( calculator.sin( Math.PI / 2 ) ).to.equal( 1 );
    } );
    it( "returns the cosine of PI", function(){
        expect( calculator.cos( Math.PI ) ).to.equal( -1 );
    } );
    it( "returns the tangent of 0", function(){
        expect( calculator.tan( 0 ) ).to.equal( 0 );
    } );
    it( "returns the logarithm of 1", function(){
        expect( calculator.log( 1 ) ).to.equal( 0 );
    } );
} );
```