**解题思路：**

1. 通过构造方法生成类
2. 在构造方法中添加属性，并为属性赋值
3. 在原型上添加方法，实现响应的功能
4. 创建对象，对方法进行测试

* 参考答案

首先创建HTML文件，创建script标签，通过构造方法生成类，在构造方法中添加属性，并为属性赋值，在原型上添加方法，实现响应的功能，创建对象，对方法进行测试。实现此作业需要按照如下步骤进行。

步骤一：新建HTML文件

首先新建一个名为index的html文件，并创建script标签。

代码如下所示：

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title></title>

</head>

<body>

<script type="text/javascript">

</script>

</body>

</html>

步骤二：创建类

通过构造方法创建类，并添加属性，为属性赋值，代码如下所示：

function Fraction (top, bottom) {

this.top = top;//分子

this.bottom = bottom;//分母

}

步骤三：完成打印方法

在Fraction的原型上添加打印方法print方法,实现思路：如果分子和分母相同，打印为1，其他情况正常打印。代码如下所示：

Fraction.prototype.print = function () {

if (this.top == this.bottom) {

console.log(1);

} else {

console.log(this.top + "/" + this.bottom);

}

}

步骤四：完成约分

在Fraction的原型上添加打印方法reduction方法,实现思路：分子和分母同时除以最大公约数，即实现约分；求最大公约数可以使用穷举法和辗转相除法。代码如下所示：

Fraction.prototype.reduction = function () {

//求最大公约数

//方式1: 穷举法

var min = Math.min(this.top, this.bottom);

var result = 1;//记录最大约数

for (var i = min; i >= 1; i--) {

if (this.top % i == 0 && this.bottom % i == 0) {

result = i;

break;

}

}

console.log(result);

//方式2: 辗转相除法

var a = this.top;

var b = this.bottom;

while (b != 0) {

var c = a % b;

a = b;

b = c;

}

console.log(a);

//约分

this.top /= a;

this.bottom /= a;

}

步骤五：完成两个分数相加

在Fraction的原型上添加打印方法add方法,实现思路：分子分母分开计算，计算完后在进行约分即可。代码如下所示：

Fraction.prototype.add = function (another) {

//两个分数: another, this

//结果分数: result

//1.创建结果分数

var result = new Fraction();

//2.计算分子, 分母

result.top = this.top \* another.bottom + another.top \* this.bottom;

result.bottom = this.bottom \* another.bottom;

//3.约分

result.reduction();

//4.返回结果

return result;

}

步骤六：完成两个分数相减，乘，除

类比步骤五，完成两个分数相减，乘，除。代码如下所示：

Fraction.prototype.minus = function (another) {

//两个分数: another, this

//结果分数: result

//1.创建结果分数

var result = new Fraction();

//2.计算分子, 分母

result.top = this.top \* another.bottom - another.top \* this.bottom;

result.bottom = this.bottom \* another.bottom;

//3.约分

result.reduction();

//4.返回结果

return result;

}

Fraction.prototype.multiply = function (another) {

//两个分数: another, this

//结果分数: result

//1.创建结果分数

var result = new Fraction();

//2.计算分子, 分母

result.top = this.top \* another.top;

result.bottom = this.bottom \* another.bottom;

//3.约分

result.reduction();

//4.返回结果

return result;

}

Fraction.prototype.divide = function (another) {

//两个分数: another, this

//结果分数: result

//1.创建结果分数

var result = new Fraction();

//2.计算分子, 分母

result.top = this.top \* another.bottom;

result.bottom = this.bottom \* another.top;

//3.约分

result.reduction();

//4.返回结果

return result;

}

步骤七：写代码进测试

创建两个分数类进行测试， 代码如下：

//创建分数对象

var num = new Fraction(12, 12);

num.print();

num.reduction();

num.print();

var arr1 = [1, 2, 3];

var arr2 = ["a", "b", "c"];

var arrAll = arr1.concat(arr2);

var num1 = new Fraction(1, 2);

var num2 = new Fraction(2, 3);

var result = num1.add(num2);

result.print();

本案例的完整代码如下所示：

代码量比较大，请查看04作业答案/01\_分数类