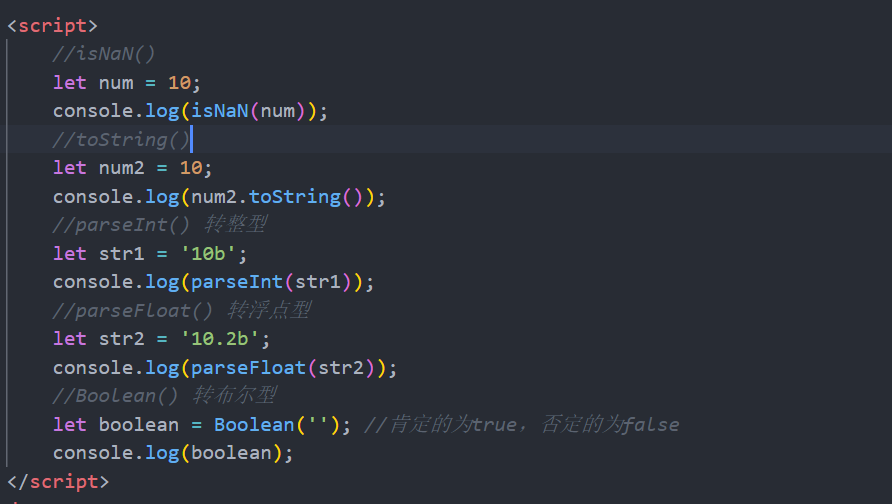
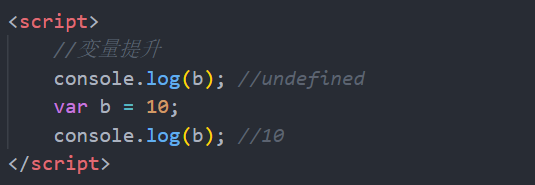
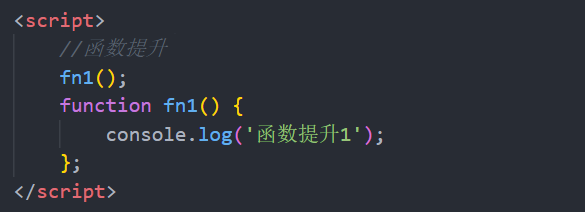
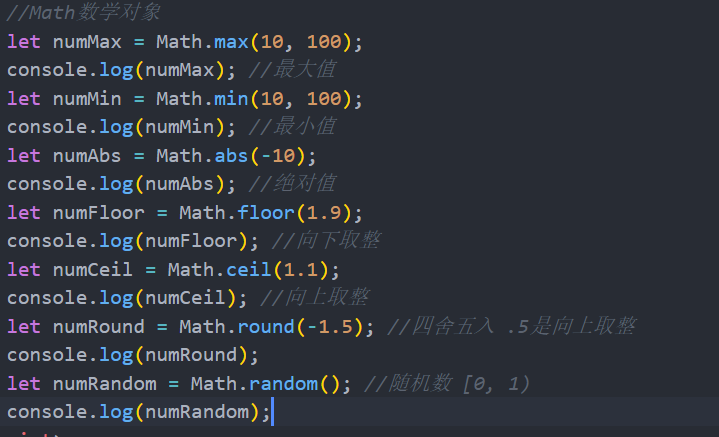
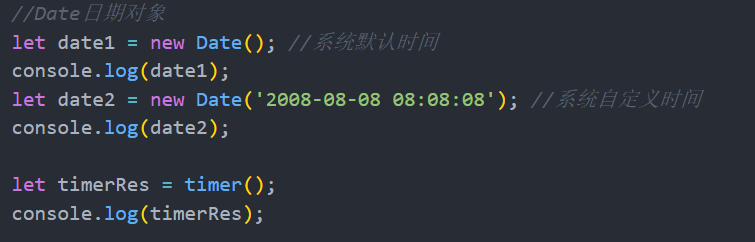
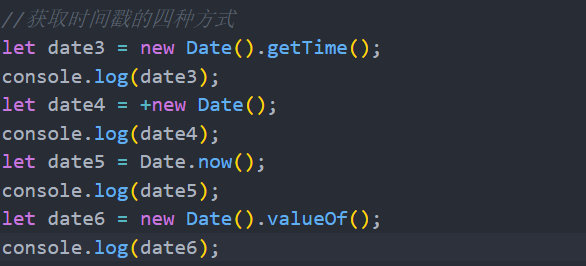
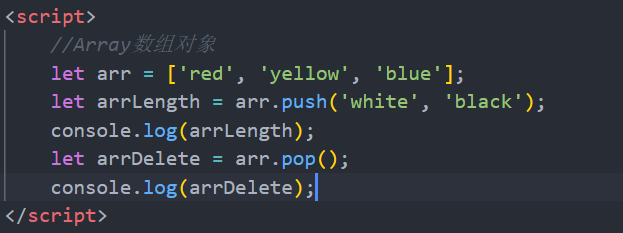
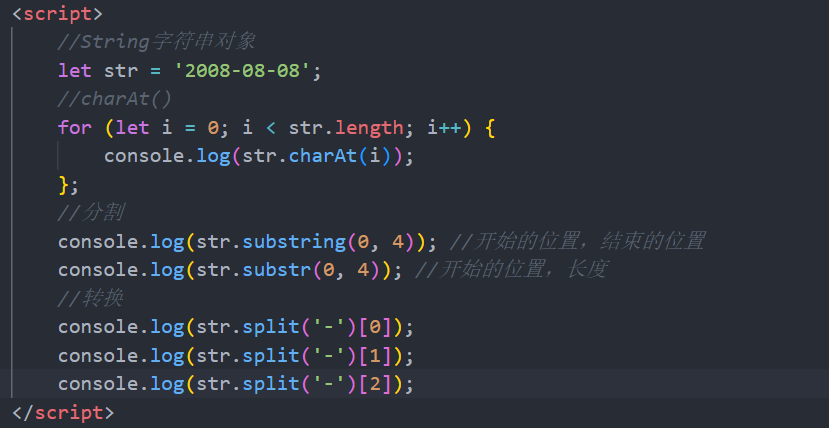
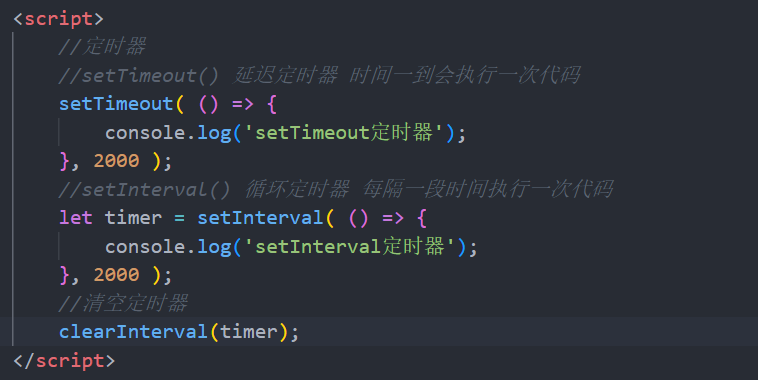
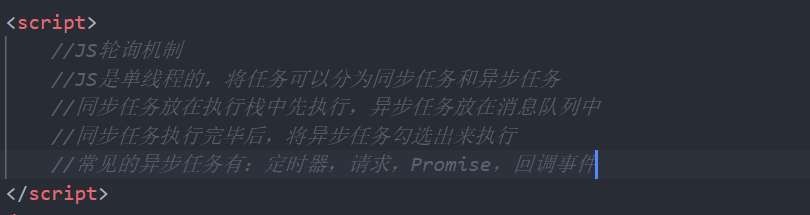
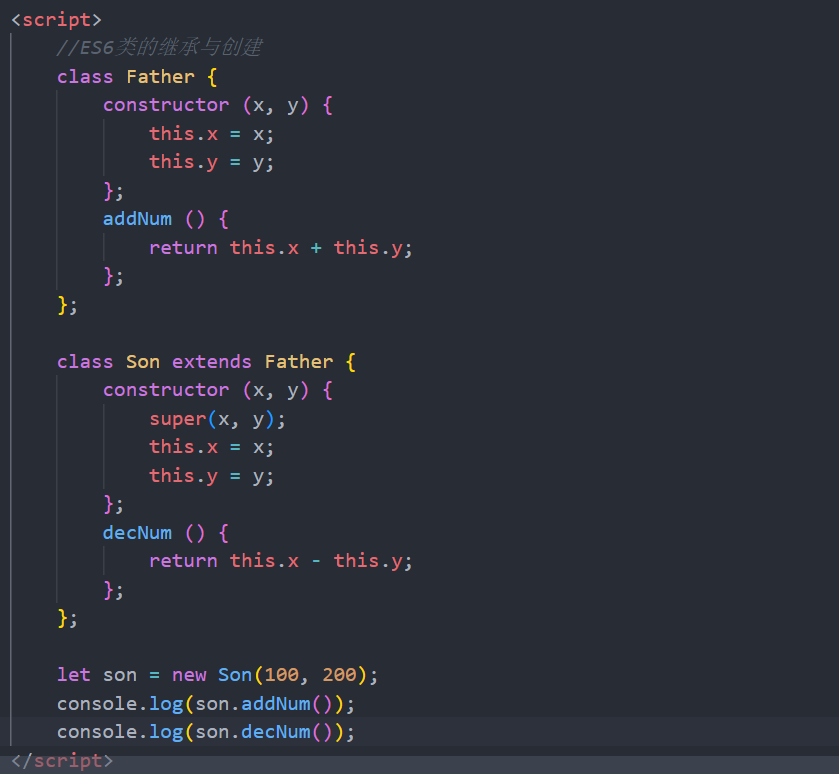
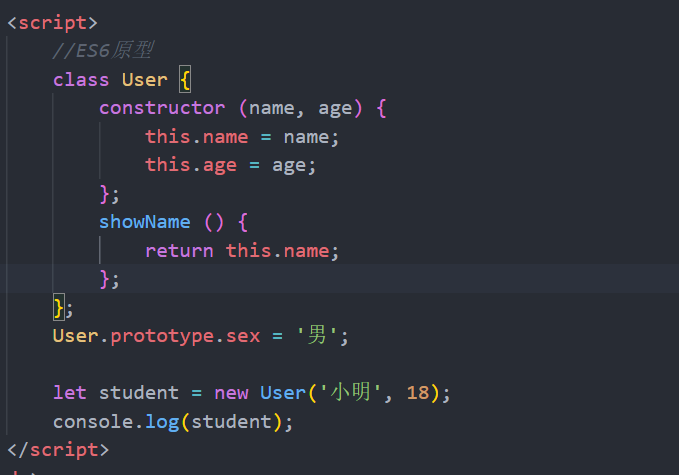
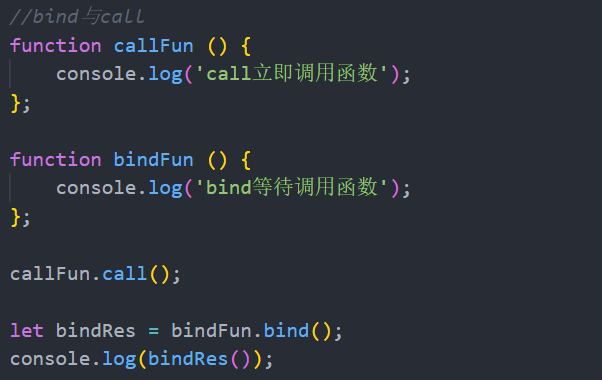
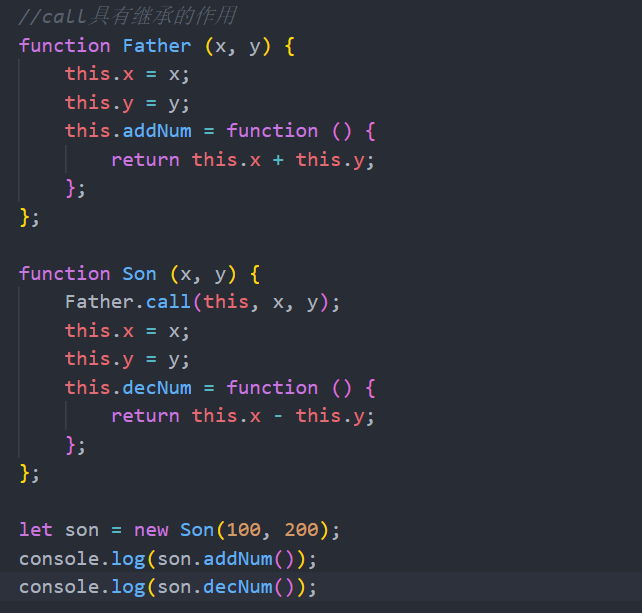
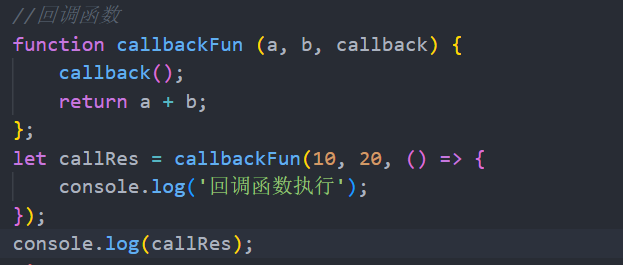
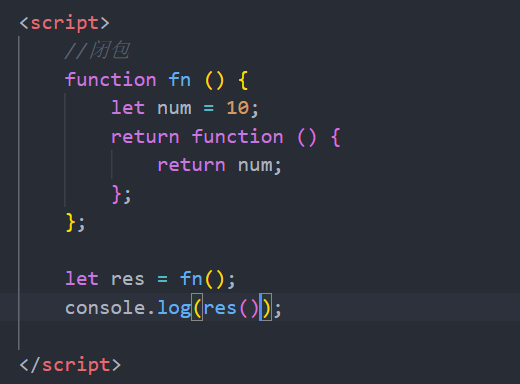
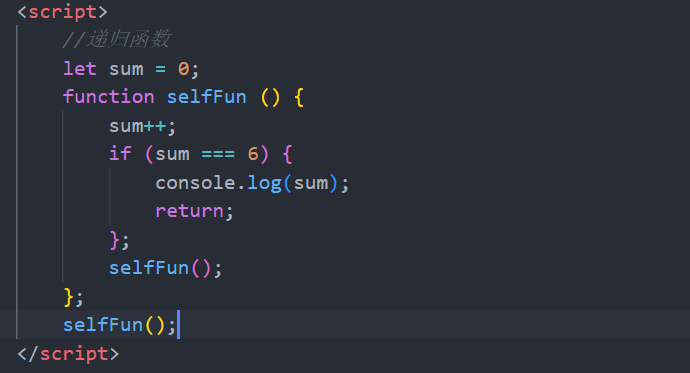
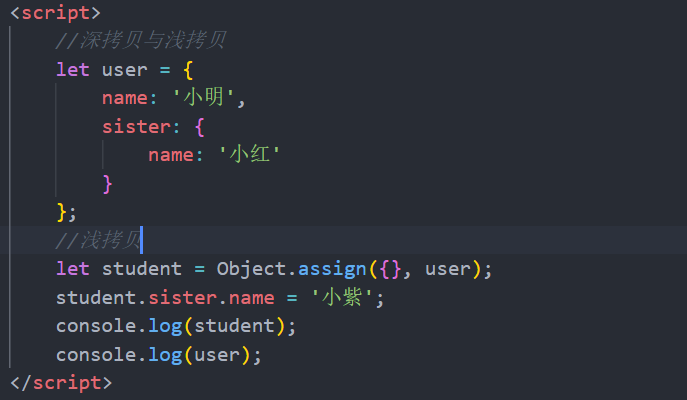
### JS-2

1. ECMAScript
   1. 变量
      1. 数据类型：基本数据类型和引用数据类型
      2. ES5基本数据类型：数值型，布尔型，字符串型，undefined，null
      3. IsNaN() 判断是否是数值型，返回false，代表是
      4. toString() 转换成字符串
      5. parseInt() 整型 parseFloat() 浮点型
      6. Boolean() 布尔型
      7. 
   2. 运算符
      1. 0.1 + 0.2 != 0.3
      2. 控制语句：顺序控制语句，流程控制语句，循环控制语句
      3. Switch流程控制语句
      4. 
   3. 循环
      1. For循环，i是计数器
      2. 浏览器的debug调试
   4. 函数
      1. 函数的值 === return返回值
      2. 函数的两种表示方式：函数声明式与函数表达式
      3. Arguments是一个伪数组，不具有数组的方法，但是具有index索引和length属性
      4. 
   5. JS引擎
      1. 浏览器为什么可以解读JS代码，是因为JS引擎，又叫做浏览器的内核
      2. 谷歌浏览器的内核是V8引擎，JS引擎又可以分为预编译和JS代码执行
      3. 预编译分为：变量提升和函数提升
      4. 
      5. 
   6. 作用域
      1. 变量：全局变量和局部变量
      2. 作用域：全局作用域和局部作用域，块级作用域
      3. 作用域链：变量取值，就近原则
   7. JS对象
      1. 访问对象的属性和方法
      2. 函数和方法有什么区别？方法是在对象里面的函数
      3. 使用new来调用构造函数
         1. 使用new来调用构造函数，内存会开辟一块空间
         2. This指向对象实例
         3. 使用new之后，不需要使用return来返回值
   8. JS内置对象
      1. Math数学对象
      2. Date日期对象
      3. Array数组对象
      4. String字符串对象
      5. 
      6. 
      7. 
      8. 
      9. 
      10. 
   9. 栈和堆
      1. 内存将空间分为栈和堆
      2. 基本数据类型存放在栈中
      3. 引用数据类型的值存放在堆中，在栈分配了内存地址，指向了堆
2. DOM文档对象模型
   1. DOM的顶级对象是document
3. BOM浏览器对象模型
   1. BOM的顶级对象是window
   2. This指向问题
   3. 定时器
   4. JS轮询机制
   5. Offset 元素位偏移 client 元素可视区 scroll 元素滚动区
   6. Mouseover与mouseenter的区别
   7. 
   8. 
   9. 
   10. 
   11. 
4. ES6类的继承与创建
   1. 
5. ES6原型
   1. 
6. 原型链
7. Bind与call
   1. Bind与call都有调用函数的作用
   2. Call是立即调用
   3. Bind是等待调用
   4. Call具有继承的作用
   5. 
   6. 
8. 高阶函数
   1. 回调函数、闭包、递归函数
   2. 回调函数：
      1. 回调函数：同步回调函数和异步回调函数
      2. 回调函数：函数作为形式参数出现在另一个函数体内
   3. 闭包：从外部作用域去访问内部的变量
      1. 作用：延申了变量的作用范围
   4. 递归函数：函数自身调用，使用return来返回值
   5. 
   6. 
   7. 
9. 深拷贝与浅拷贝
   1. 
   2. 
10. 正则表达式
    1. /^$/ 边界符 以...开始 以...结束
    2. /[0-9a-zA-Z\_]{3,6}/ 字符组合
    3. 量词符 \* 表示出现0到无数次 + 表示出现1到无数次 ? 表示出现0或者1次
    4. 预定义类 \d 数字 \D 取反 \w [0-9a-zA-Z\_] \W 取反 \s 空格或者回车 \S 取反