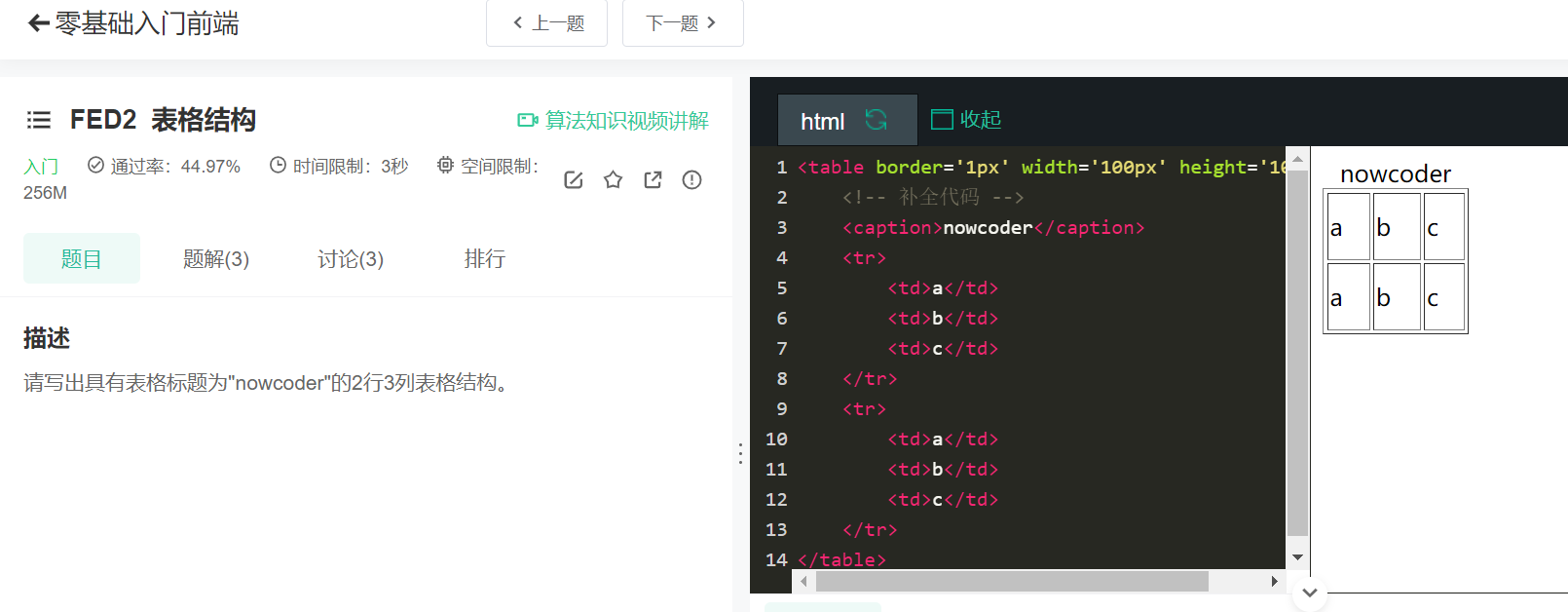
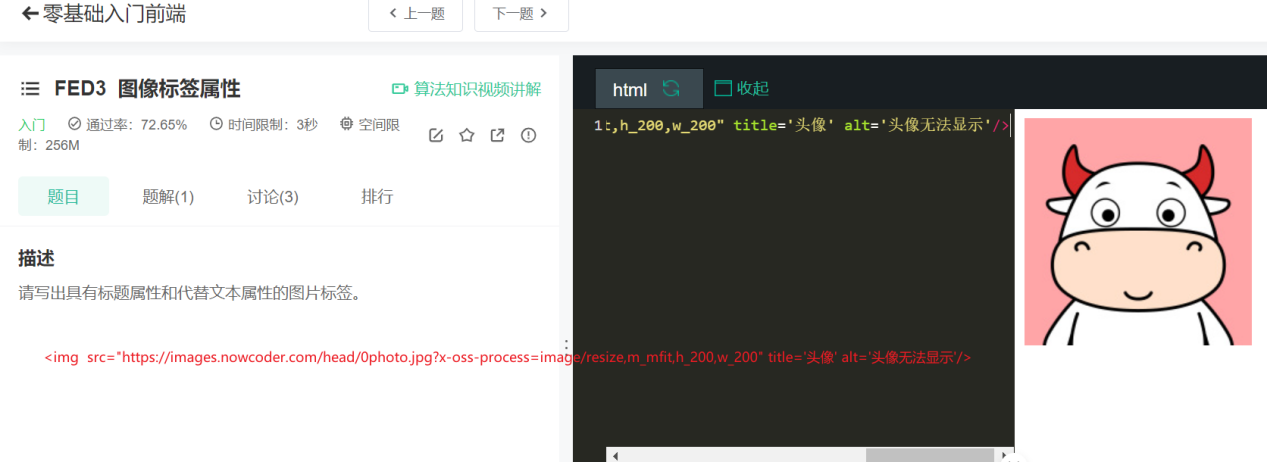
## caption



## alt



## target



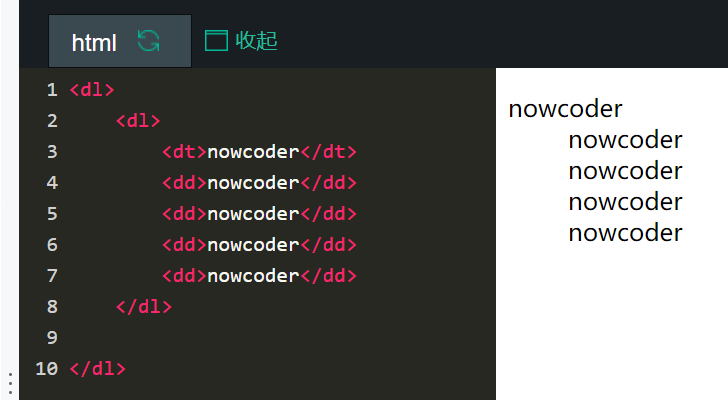
## dl dt dd

dl definition list 定义一个列表

dt definition title 定义一个标题

dd definition description 定义一个描述

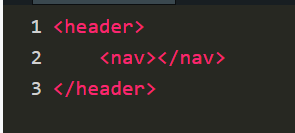
请写出列表项为"nowcoder"且列表项内容也为"nowcoder"的自定义列表。



## H5语义化标签

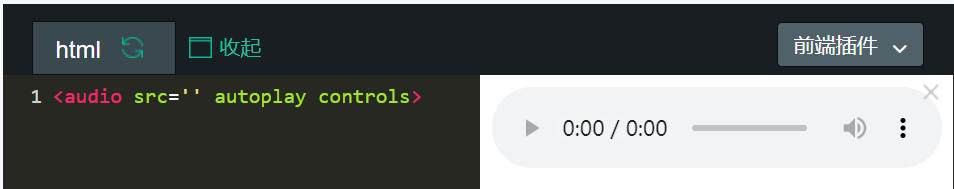
请使用**语义化标签**创建头部标签且包含导航标签。  
注意：只需在html模块填写标签结构，有且仅有一个头部标签和一个导航标签。

(语义化标签是H5新标签）



### audio controls

请写出具有控件功能的音频媒体标签。

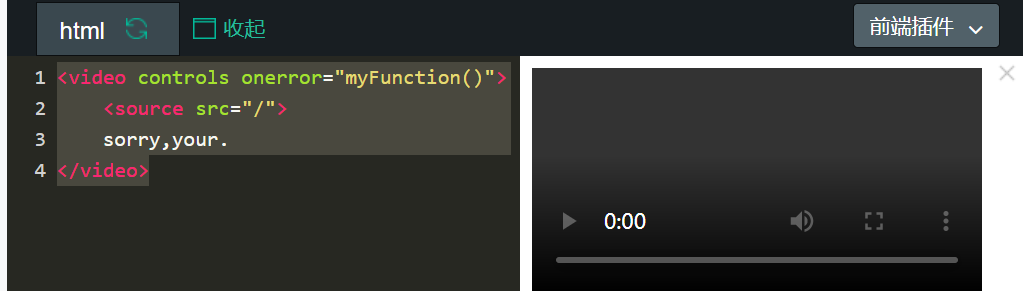


controls会显示出控件

### video onerror controls

**视频媒体标签属性**

请写出具有当视频的媒体数据加载期间发生错误时执行某个方法事件的视频媒体标签。



# CSS

请将html模块中字体内容是"红色"的字体颜色设置为"rgb(255, 0, 0)"，"绿色"设置为"rgb(0, 128, 0)"，"黑色"设置为"rgb(0, 0, 0)"，且字体大小都为20px。



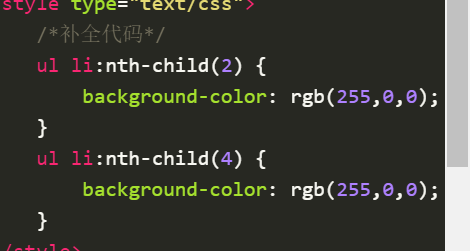
## CSS选择器——伪类选择器 :

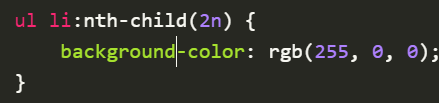
请将html模块中ul列表的第2个li标签和第4个li标签的背景颜色设置成"rgb(255, 0, 0)"。

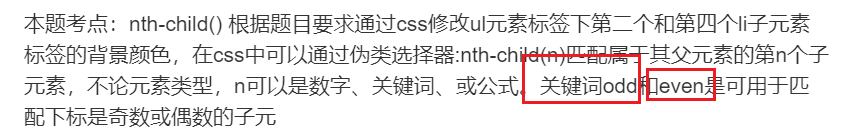
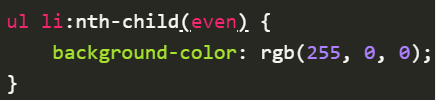
用例输入：

function () { var first = getComputedStyle(document.querySelectorAll('li')[1]).backgroundColor == "rgb(255, 0, 0)"; var result = first ; return result; }

解答：

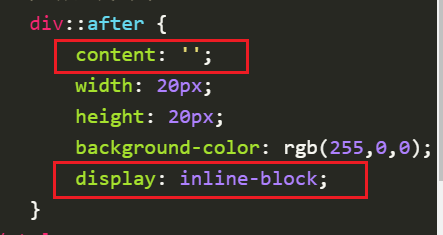






## CSS选择器——伪元素 ::

请给html模块的div元素加一个后伪元素，且后伪元素的宽度和高度都是20px，背景颜色为"rgb(255, 0, 0)"。

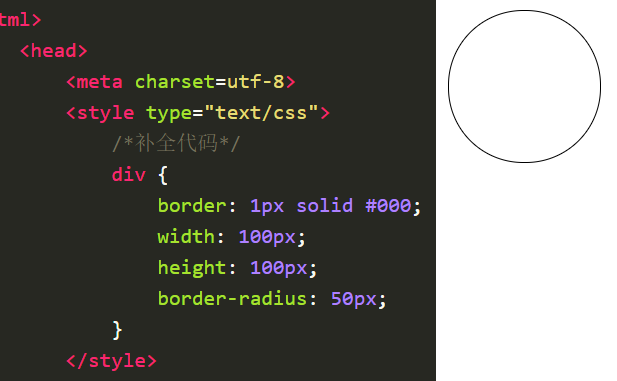


content必不可少

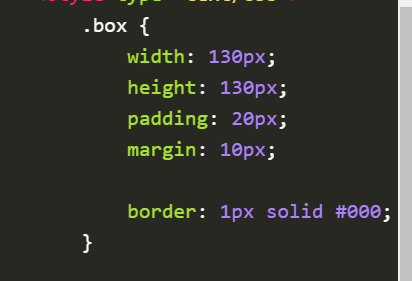
display设置为块或行内块才可以设置宽高。

## 按要求写一个圆

请将html模块的div元素设置为一个半径是50px的圆，且边框为1px的黑色实线。  
要求：  
1. 圆角属性仅设置一个值  
2. 圆角属性单位请使用px  
注意：由于圆角属性设置广泛且都可以实现题目效果，所以请按照要求规范书写。



请将html模块类为"box"的div元素宽度和高度都设置为100px，且内间距为20px、外间距为10px。

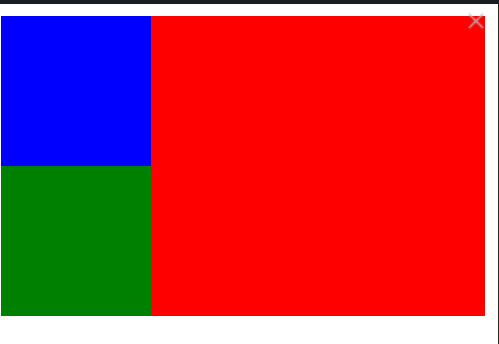


内外边距和边框都会改变盒子

## 浮动和清除浮动

请将类为"left"的div元素和类为"right"的div元素在同一行上向左浮动，且清除类为"wrap"的父级div元素内部的浮动。

原来布局：



若没有文字高度的支撑，不考虑边框，父级元素高度会塌缩成零。浮动元素甚至影响到了其父元素的兄弟元素.



清除浮动顾名思义：清楚子元素浮动所带来的父元素消失的问题。

解决方法：<https://blog.csdn.net/u012207345/article/details/78279961>

1. clear: both; clear: left;clear right;
2. 利用块级元素,利用为元素：after，都和1的原理相同。
3. overflow:hidden 块格式化上下文。

原理：

当元素设置了overflow样式，且值不为visible时，该元素就建构了一个BFC(哪些情况下，元素可以建构出BFC，可以看查看CSS文档对[BFC的定义](https://link.juejin.im/?target=https://www.w3.org/TR/CSS2/visuren.html%23block-formatting" \t "https://blog.csdn.net/u012207345/article/details/_blank))。在我们的例子中，.topDiv因设置了值为auto的overflow样式，所以该元素建构出一个BFC，按照第三个特点，BFC的高度是要包括浮动元素的，所以.topDiv的高度被撑起来，达到了清除浮动影响的目的。(至于为什么值为visible的overflow不能建构BFC，[这个答案](https://link.juejin.im/?target=https://stackoverflow.com/questions/9943503/why-does-css2-1-define-overflow-values-other-than-visible-to-establish-a-new-b?answertab=votes%23tab-top" \t "https://blog.csdn.net/u012207345/article/details/_blank)给了一个解释)

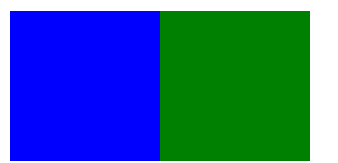
其实，这里overflow的作用就是为了构建一个BFC区域，让内部浮动的影响都得以“内化”。如果你看了[BFC的定义](https://link.juejin.im/?target=https://www.w3.org/TR/CSS2/visuren.html%23block-formatting" \t "https://blog.csdn.net/u012207345/article/details/_blank)，你会发现，构建一个BFC区域的方法有很多种，overflow只是其中的一种，那在这里，我们是否也可以利用其它的方式构建BFC，且同样能达到清除浮动的目的呢？

BFC定义中说，inline-block同样也能构建BFC

1. 给父元素加height

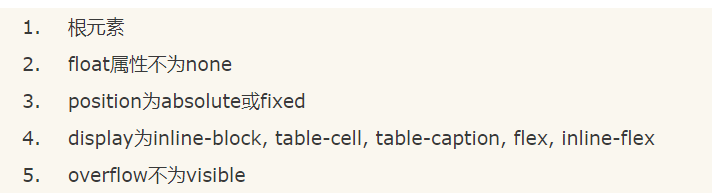


1. 给父元素加 position: absolute;

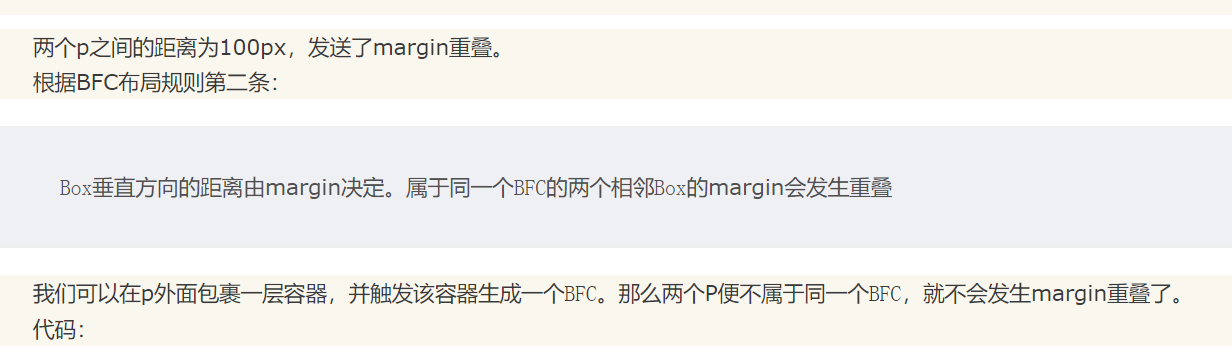


CSS的BFC详解：<https://blog.csdn.net/shadow_zed/article/details/72811975>

可以生成BFC的CSS样式：



还可以清楚塌陷：margin重叠



 .wrap {

         overflow: hidden;

     }

## 固定定位

请将html模块类为"box"的div元素固定在视口的左上角。

.box {

width: 100px;

height: 100px;

/\*补全代码\*/

position: fixed;

top: 0;

left: 0;

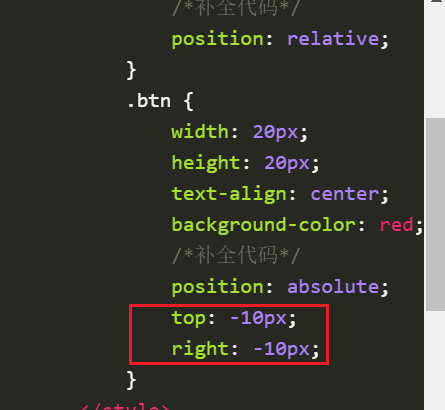
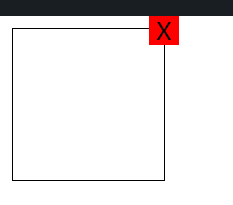
background-color: #000;

}

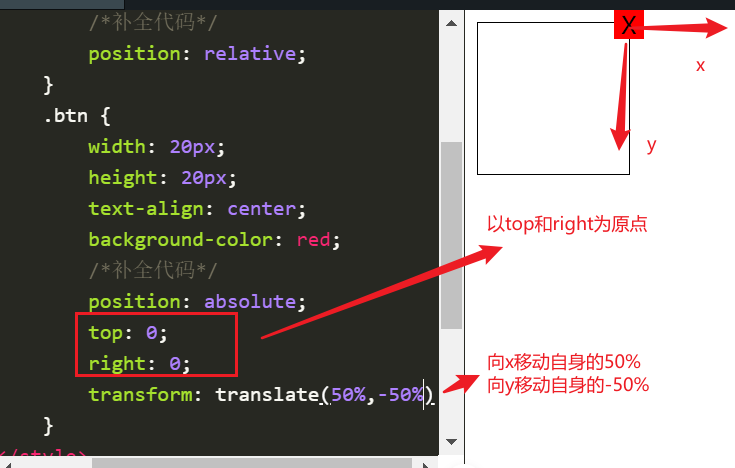
## 绝对定位

请将html模块类为"btn"的div元素中心定位在类为"wrap"的父级div元素右上顶点处。

子绝父相



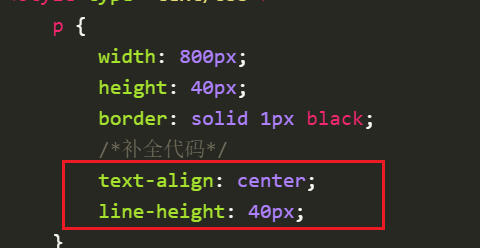
另一种方法：



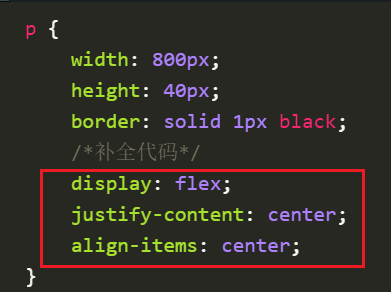
## 行内元素垂直水平居中

**因为是行内元素，且是文本元素，所以可以text-align**

**有些是块级元素，需要margin： 0 auto；**



### 借助flex布局：



align-items是单行。

## em



## rem



**html { .box { 此时.box中所有用rem单位的，**

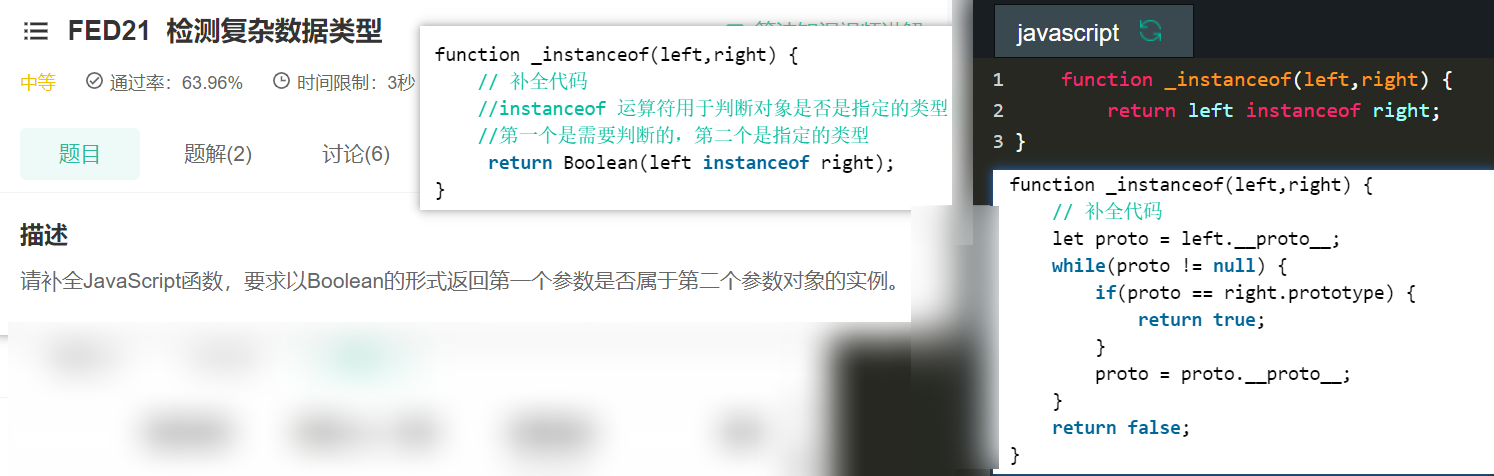
**font-size: 1px; 2rem; 都是相对于html中的1px而言的。**

**} }**

## typeof



## instanceof

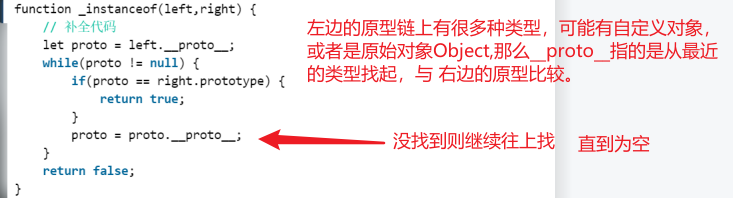


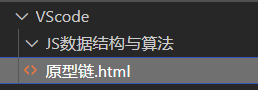
instanceof主要是用于判断某个实例是否属于某个类型，

原型和原型链需要区分开：

原型prototype

原型链\_\_ptoto\_\_





## 字符串拼接OR数组添加删除元素



splice() 方法向/从数组添加/删除项目，并返回删除的项目。

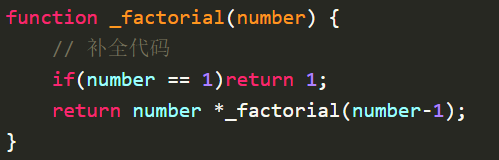
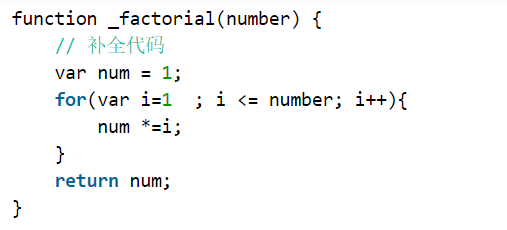
**注释：**splice() 方法会改变原始数组。

// splice是添加或删除数组的，对字符串无用，而且会改变原数组，会返回被删除的字符串

## 待续

## 阶乘





## 绝对值

请补全JavaScript函数，要求返回数字参数的绝对值。

function \_abs(number) {

// 补全代码

let n = number;

if(n < 0) n = -n;

return n;

}

function \_abs(number) {

// 补全代码

return number>=0?number:-number

}

function \_abs(number) {

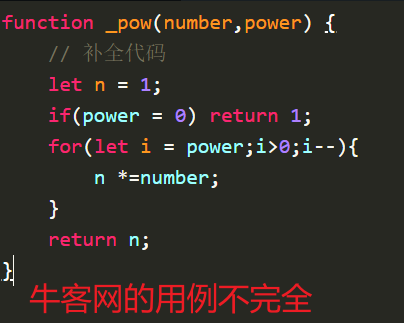
// 补全代码

//JS Math对象方法 abs() 求绝对值

return Math.abs(number);

}

请补全JavaScript函数，要求返回底数为第一个参数，幂为第二个参数的数值。



function \_pow(number,power) {

// 补全代码

return Math.pow(number,power);

}

**平方根**

**function \_sqrt(number) {**

**return Math.sqrt(number);**

**}**

**只限平方根为整数的情况：**

**function \_sqrt(number) {**

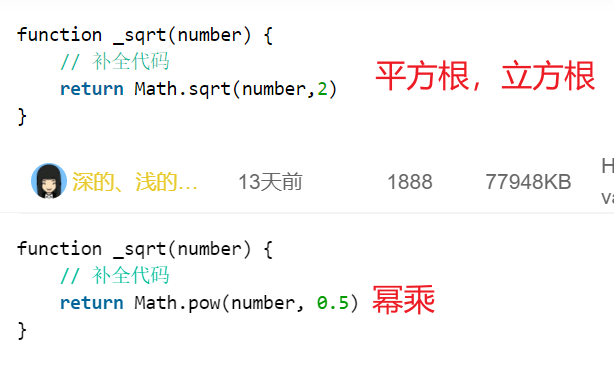
**// 补全代码**

**for (let i = 1; i < number/2; i++) {**

**if ((i \* i) == number) return i**

**}**

**}**



## 余数

请补全JavaScript函数，要求返回数字参数除以2的余数。

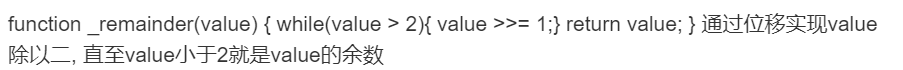
value%2;

return value - parseInt(value / 2)

return value - (value>>1)\*2

value - (value>>2) 这个不对啊..为啥大家都是这个答案- -



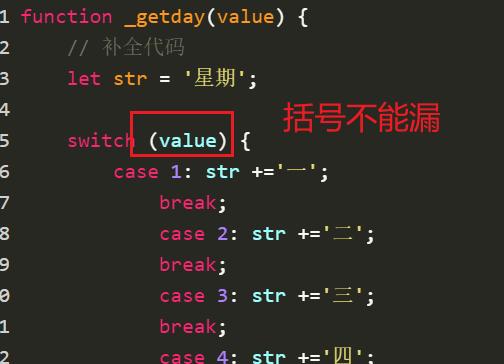
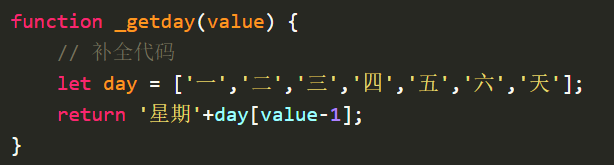


　>>，有符号右移位，将运算数的二进制整体右移指定位数，整数高位用0补齐，负数高位用1补齐（保持负数符号不变）。

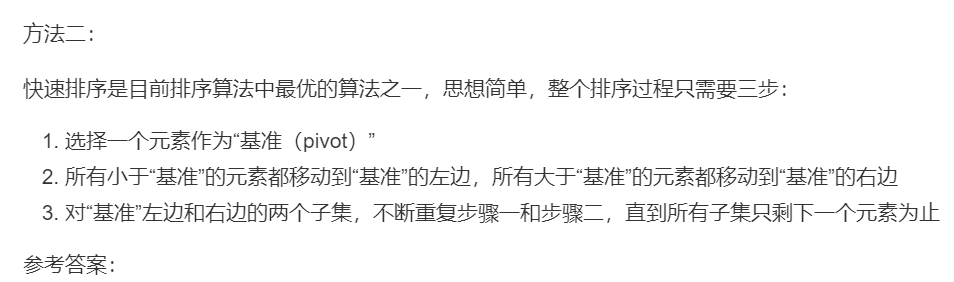
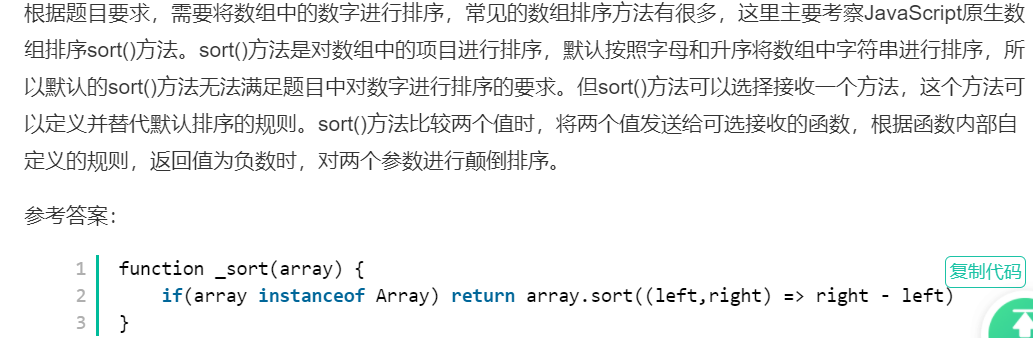
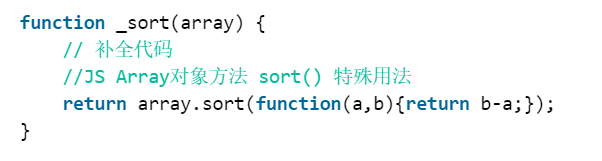
## 返回星期数

## **描述**

请补全JavaScript函数，要求以字符串的形式返回数字参数所对应的星期。  
示例：  
1. \_getday(1) -> "星期一"  
2. \_getday(7) -> "星期天"



## 排序：



function quickSort(array) {

　　if (array.length <= 1) return array

　　let pivotIndex = Math.floor(array.length / 2)

　　let pivot = array.splice(pivotIndex, 1)[0]

　　let left = []

　　let right = []

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

　　　　if (array[i] < pivot) {

　　　　　　left.push(array[i])

　　　　} else {

　　　　　　right.push(array[i]);

　　　　}

　　}

　　return quickSort(left).concat([pivot], quickSort(right))

}

function \_sort(array) {

// 补全代码

// 复习一下冒泡

for(var i = 0; i < array.length-1; i++){

var flag = true

for(var j = array.length; j > i; j--){

if(array[j] > array[j-1]){

var temp = array[j]

array[j] = array[j-1]

array[j-1] = temp

flag = !flag

}

}

if(flag) return array

}

return array

}

## 字符串大写

请补全JavaScript函数，要求将字符串参数转换为大写字符串并返回。

function \_touppercase(string) {

return string.toUpperCase();

}

function \_touppercase(string) {

const arr = []

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

const codePoint = string.codePointAt(i)

if(codePoint >= 97 && codePoint <= 122) {

const charactor = String.fromCodePoint(codePoint - 32)

arr[i] = charactor

} else {

arr[i] = string.charAt(i)

}

}

return arr.join('')

}

function \_touppercase(string) {

function uppercase(char, i) {

if(char >= 'a' && char <= 'z') return String.fromCodePoint(string.codePointAt(i) - 32)

return char

}

return Array.from(string).map(uppercase).join('')

}

function \_touppercase(string) {

const arr = []

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

const codePoint = string.codePointAt(i)

if(codePoint >= 97 && codePoint <= 122) {

const charactor = String.fromCodePoint(codePoint - 32)

arr[i] = charactor

} else {

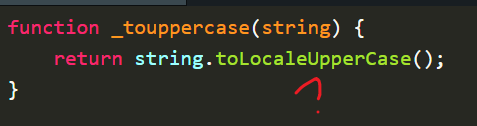
arr[i] = string.charAt(i)

}

}

return arr.join('')

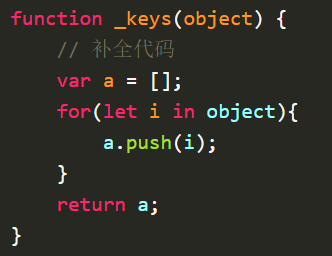
}



## 待续2

## 对象属性键名

请补全JavaScript函数，要求以数组的形式输出对象各个属性的键名。  
示例：  
1. \_keys({name:'nowcoder',age:7}) -> ['name','age']  
注意：只需考虑对象属性均为原始数据类型的情况。



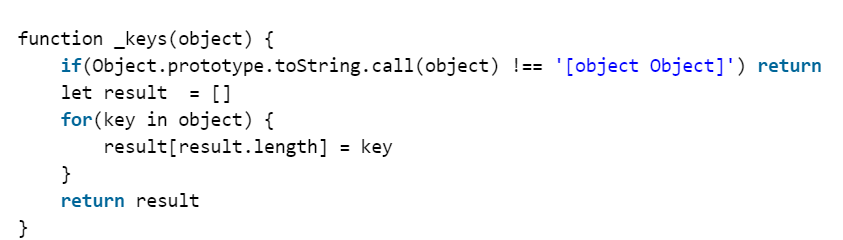
a.push

let i in object i 属性名 object[i] 属性值

Object.keys(object)

 Object.getOwnPropertyNames(object)

Reflect.ownKeys(object)



## 对象数字

请补全JavaScript函数，要求将数字参数转换为对象的形式返回。  
示例：  
1. typeof number === 'number' -> typeof \_numbertoobject(number) === 'object'

function \_numbertoobject(number) {

// 补全代码

return new Number(number);

}

function \_numbertoobject(number) {

// 补全代码

return {number};

}

## 去除字符串两端空格

①正则表达式，以空格开头或结尾，空格\s

return string.replace(/(^\s\*)|(\s\*$)/g,'')

不要忘记括号()，所有\*，以及或|

②ASCII码匹配，直接与’ ‘空格比较。

③string.trim( )，去掉字符串两端所有ASCII小于等于空格即****32（\u0020）的字符****

****trim( )源码修改过后的：****

****function \_trim(string) {****

****// 补全代码****

****var len = string.length;****

****var st = 0;****

****var val = string;****

****//char[] val = value; /\* avoid getfield opcode \*/****

****while ((st < len) && (val[st] <= ' ')) {****

****st++;****

****}****

****while ((st < len) && (val[len - 1] <= ' ')) {****

****len--;****

****}****

****return ((st > 0) || (len < string.length)) ? string.substring(st, len) : this;****

****}****

## 输出日期

请补全JavaScript函数，要求以字符串的形式输出时间戳参数所对应的"年-月-日"。  
示例：  
1. \_date(1631159776311) -> '2021-9-9'

function \_date(number) {

// 补全代码

let date = new Date(number);

let year = date.getFullYear();

let month = date.getMonth() + 1;

let day = date.getDate();

return year + '-' + month + '-' + day;

}

## 数字取整

请补全JavaScript函数，要求返回数字参数的整数部分。

function \_int(value) {

// 补全代码

//return Math.floor(value);

//return Number((''+value).split('.')[0]);

//return +((''+value).split('.')[0]);

//return value >> 0;

//return parseInt(value);

}

## 数组反转

请补全JavaScript函数，要求将参数数组反转之后返回。

function \_reverse(array) {

// 补全代码

let arr = [];

for(let i = array.length-1;i>=0;i--){

arr[arr.length] = array[i];

}

return arr;

}

function \_reverse(array) {

for(let i = array.length-1,j = 0;j<i;i--,j++){

let tmp = array[i];

array[i] = array[j];

array[j] = tmp;

}

return array;

//return array.reverse()

}

## 数组转字符串

请补全JavaScript函数，要求将参数数组转换为字符串输出。  
示例：  
1. \_join([1,'2',3]) -> "123"  
注意：只需考虑一维数组且数据类型为原始数据类型。

# JS



变为提升，但函数表达式没有提升。

