bilibili

1、下面哪几个和 `http://store.company.com/dir/page.html`

符合同源策略?

A、 http://store.company.com/dir2/other.htm

B、 https://store.company.com/dir/secure.html

C、 http://store.company.com:81/dir/etc.html

D、 http://news.company.com/dir/other.html

正确答案: A

2、关于DOMContentLoaded和load事件说法正确的是?

A、 DOMContentLoaded事件比load事件更早执行

B、 load事件比DOMContentLoaded事件更早执行

C、 按监听代码从上到下先后执行

D、 dom文档完全加载完后执行load事件

正确答案: A

> 1. DOMContentLoaded
> 当纯HTML被完全加载以及解析时，DOMContentLoaded事件会被触发，而不必等待样式表，图片或者子框架完成加载。
>
> 2. Load
> 当一个资源及其依赖资源已完成加载时，将触发load事件。
>
> 3. 在任何情况下，DOMContentLoaded 的触发不需要等待图片等其他资源加载完成。

> 页面上所有的资源（图片，音频，视频等）被加载以后才会触发load事件

3、如何在 div 容器里展示 这几个字符?

A、
B、 ” ”
C、document.querySelector(’div’).innerText = ” ”
D、document.querySelector(’div’).innerHTML = ” sssssss ”

正确答案: C

> innerHTML：设置或获取标签所包含的HTML与文本信息。（不含标签本身）
>
> innerText：设置或获取标签所包含的文本信息。（不含标签本身）
>
> outerHTML：设置或获取标签本身以及所包含的HTML与文本信息。（包含标签本身）
>
> outerText：设置或获取标签本身以及所包含的文本信息。（包含标签本身）
>
> 例如：
>
> `
>
> this is text
>
> <script>
> var div=document.getElementsByTagName(”div”);
> console.log(div[0].innerHTML); // this is text
> console.log(div[0].innerText); // this is text
> console.log(div[0].outerHTML); // this is text
>
> console.log(div[0].outerText); // this is text
> </script>
> `

4、有n级台阶，每一步可以走1级或2级，问一共有多少种走法

正确答案

```
function NumberOf1(number) {
  // write code here
  var arr = [1, 2]
  for (var i = 2; i < number; i++) {
    arr[i] = arr[i - 1] + arr[i - 2]
```

```
    }
    return arr[number - 1]
  }
```

参考解析

共n个楼梯，第一次跳1阶，后面还有n-1，有n-1种跳法；第一次跳两个台阶，后面还有n-2，有n-2种跳法。所以递归方式是f(n-1)+f(n-2)

```
`js
function getStairs(stair) {
 if(stair <= 0) {
    return0
 }
 if(stair === 1) {
    return1
 }
 if(stair === 2) {
    return2
 }

 return getStairs(stair - 1) + getStairs(stair - 2)
}

conststair = parseInt(readline())
console.log(getStairs(stair))
`
```

5、判断由”()[]{}”6种括号组成的字符串是否合法

1. 所有括号必须闭合

2. 左括号必须在正确的位置闭合

正确答案

```
function check(str) {
  const ary = [];
  const map = {
    '[': ']',
    '{': '}',
    '(': ')',
  };
```

```
  for(let v of str) {
    if(v === '['|| v === '{'|| v === '(') {
      ary.push(v);
    } else{
      if(map[ary.pop()] !== v) {
        return false;
      }
    }
  }
  if(ary.length === 0) {
    return true;
  }
  return false;
}
```

参考解析

1.利用 map 实现一一对应

2.利用栈

、

```
function IsValue(str) {
    let map = new Map([
        ['(',')'],
        ['{','}'],
        ['[',']']
    ])
    let stack = [];
    for(let i = 0 ; i < str.length ; i ++) {
        if(map.has(str[i])) { // 在左边
            stack.push(str[i])
        } else { // 在右边
            if(stack.length == 0) return false;
            let top = stack[stack.length - 1]; // 栈顶元素
            if(map.get(top) == str[i]) {
                stack.pop()
            } else {
                return false;
            }
        }
    }
    return true;
}

while(line = readline()) {
    var str = line;
    print(IsValue(str))
```

```
}
`
```

6、找出有序数组（从小到大排列）中和为sum的两个数，要求复杂度为O(n)，找到一组即可

正确答案

```
function solve (arr, target) {
  let i = 0
  let j = arr.length - 1
  let first, last

  while (i < j) {
    first = arr[i]
    last = arr[j]
    const sum = first + last
    if (sum === target) return [first, last]
    else if (sum > target) j--
    else i++
  }

  return null
}
```

参考答案1

```
`js
var n = parseInt(readline());
var line = readline().split(" ");
var arr = [];
for(var i=0;i<n;i++){
    arri);
}
var sum = parseInt(readline());
var left = 0,right = n-1;
var str = "";
while(left<right){
    var temp = arrleft;
    if(temp<sum){
        left++;
    }else if(temp>sum){
        right--;
    }else{
        str = arrleft;
```

```
            break;
        }
    }
    if(str){
        console.log(str);
    }else{
        console.log(”notfound”);
    }
    `
```

参考答案2

实现思路：用一个对象的key来存储遍历过的数字（value值任意，这里用的是0），每次循环只需判断(sum−当前值)的key值对应的value是不是为0（为0就代表之前遍历过这个数字）

因为只用到一个for循环，时间复杂度为O(n)

```
`
let length = parseInt(readline()),
    arg = readline().split(” ”),
    sum = parseInt(readline()),
    temp = {},// 用来存储遍历过的数字的对象
     flag = 0;

for(let i = 0; i < length; i++) {
    let a = arg[i];
    let b = sum − a;
    temp[a] = 0;
    if(temp[b] === 0) {
        console.log(b+” ”+a);
        flag = 1;
        break;
    }
}
if(flag === 0) {
    console.log(”notfound”);
}
`
```

7、获取列表中战队名是BLG 位置上路的 选手对象?

列表：

```
const players = [ {name: 'UZI', team: 'RNG', position: 'ADC'},
{name: 'theshy', team: 'IG', position: 'TOP'},
{name: 'Metoer', team: 'BLG', position: 'Jungle'},
{name: 'ADD', team: 'BLG', position: 'TOP'},
{name: 'Scout', team: 'EDG', position: 'Middle'},
{name: 'iBoy', team: 'EDG', position: 'ADC'},
{name: 'Baolan', team: 'IG', position: 'Support'},
{name: 'Xiaohu', team: 'RNG', position: 'Middle'}]
```

选项

A、players.filter(x=> x.position === 'TOP' && x.team === 'BLG')
B、players.get(position='TOP', team='BLG')
C、players.find(position='TOP', team='BLG')
D、players.findOne(position='TOP', team='BLG')

正确答案: A

8、以下代码运行结果

```
function setname(name){
 this.name = name
}
setname.prototype.printName = function(){ console.log(this.name) }
let a = new setname("cc")
a.name = "dd"
a.__proto__.name = "ee"

a.__proto__.printName()  // ?
a.printName() // ?
```

选项

A、ee dd
B、cc dd
C、ee cc
D、ee Error

正确答案: A