**Data-内容-前端 外包笔试题目**

友情提示

* 加分项：编程风格优雅，包括但不限于清晰的逻辑及代码格式、易懂的变量命名、可扩展性强等
* 若无法作答，可体现思路想法

**HTML + CSS**

1. 下面代码在页面中呈现什么内容，为什么？

|  |
| --- |
| HTML  <!-- html -->  <div class="div" id="div" style="visibility:hidden; color: #fff;">  我是一个div标签  </div>  <span>你猜下我的位置在哪</span>  <!-- css -->  <style>  div { display: none; width: 100px; height: 20px; }  .div { display: inline }  #div { color: red }  </style>  1这个代码行内优先级最高 其次使id class 优先执行行内的属性  2 class类 的属性dispaly使块级变成了内联 不支持宽高,  3 虽然变成了内联元素的div的内容不显示 但是里面文字撑开了空间 因此span标签显示在一个看不见的的空白的右边 |

1. 实现两栏布局，左边一栏宽度固定 200px，右边一栏宽度自适应

如果能用多种方式实现，加分

|  |
| --- |
| HTML  <!-- html -->  <div class="container">  <div class="left">这是左边栏</div>  <div class="right">这是右边栏</div>  </div>  <!-- css -->  <style>  <!-- todo -->  .container {  width:500px  display:flex;  justify-content: space-between;    }    .left {  .left{ width:200px }  }    .right: {  flex:1  }  </style> |
|  |
|  |
|  |

**JavaScript**

以下两道题**如果只写出运行结果，没有分数**，需要**解释原因**

1. 请写出下面这段自执行函数代码运行时输出的先后顺序，并解释原因

|  |
| --- |
| JavaScript  (() => {  console.log(1);    setTimeout(() => {  console.log(2);  }, 0);    new Promise((resolve, reject) => {  resolve()  console.log(3);  }).then(() => {  console.log(4);  });    console.log(5);  })();  这里面有一次性定时器 和Promise两个异步函数  主程序先走同步 后执行异步  同步执行的时候 遵循顺序执行结构 因此最先输出 1 然后是5  一次性定时器设置的时间是0秒 又在promise前面 所以先执行 输出2  Promise 触发了relolve成功回调 依次输入 4 3  最后顺序是 1 5 2 4 3 |

1. 请写出下面代码运行时输出的信息，并解释原因

|  |
| --- |
| JavaScript  function A(){  let person = {  name: "张三",  age: 10,  getName: function() {  console.log(this.name); 张三  },  getAge: () => {  console.log(this.age); 10  }  }  return person;  }  function B(){  this.name = "李四";  this.age = 20;  }  // 以下每一行代码执行，会输出什么内容  A().getName() 张三  A().getAge() 10  对象嵌套个函数 属于闭包 函数可以访问对象和函数体之间的变量  B() 函数体 {this.name:’李四’,this.age=20}  构造函数B() 没有被调用 所以是函数体  A().getName() 张三  A().getAge() 10  对象嵌套个函数 属于闭包 函数可以访问对象和函数体之间的变量 |

**编程题**

1. 数组平铺，实现一个函数，将数组中的数组打平成一维数组

|  |
| --- |
| JavaScript  /\*\*  eg：input: [[5, 7, 4], 3, [89, [8, 9]]]  ouput: [5, 7, 4, 3, 89, 8, 9]  \*\*/  // 不可以使用 Array.prototype.flat() 来实现  **Array.prototype.isPrototypeOf(obj);**  function flat(arr) {  let newArr = [ ];  for(let I =0;i<arr.length;i++){  if(Array.prototype.isPrototypeOf(arr[i])){  for(let j = 0; j<arr[i].length;j++){  newArr.push(arr[i].[j]);  }  }else{  newArr.push(arr[i]);  }  retrun newArr  }  } |
|  |
|  |
|  |

1. 树的每一个节点包含了 id 和 label 字段，实现一个函数，输入关键字 searchKey，用于树节点 id 的匹配，查找 searchKey 对应树节点的 label，如果查找不到则返回 null

|  |
| --- |
| JavaScript  // 定义一个节点 node = { id, label, children: [node, node] }  // 如果是叶子节点， children = undefined  // node 是根节点  function searchTree(node，searchKey) {    } |

1. 【加分题】请按要求实现 func 函数

|  |
| --- |
| JavaScript  const sleep = ms => new Promise(resolve => setTimeout(resolve, ms))  const func = (promiseFuncs) => {  // todo: 实现 func  }  func([  () => console.log('start'),  () => sleep(1000),  () => console.log('1'),  () => sleep(2000),  () => console.log('2'),  () => sleep(3000),  () => console.log('end'),  ])  // 先输出 start  // 一秒后输出 1  // 再过两秒输出 2  // 再过三秒输出 end |

1. 【加分题】实现一个栈数据结构，接口中实现以下 4 个函数
2. in: 数据进栈
3. out: 数据出栈
4. top: 返回栈顶的数据
5. size: 返回栈数据的长度

|  |
| --- |
| JavaScript  class Stack {  constructor() {}  in(value) {  // 数据进栈  // todo: 你的代码  this.value=value;  }  out() {  // 数据出栈  // todo: 你的代码  in(x)  }  top() {  // 返回栈顶的数据  // todo: 你的代码  this.in(value)  }  size() {  // 返回栈数据的长度  // todo: 你的代码  this.in(val1,val2,val3)    }  }  // 要求当执行下列代码时，能输出预期的结果  const stack = new Stack()  stack.in('x')  stack.in('y')  stack.in('z')  stack.top() // 输出 'z'  stack.size() // 输出 3  stack.out() // 输出 'z'  stack.top() // 输出 'y'  stack.size() // 输出 2 |
|  |
|  |