

实验一 ECMAScript 基础

一、实验目的

1. 掌握 ECMAScript 6 的基本语法；
2. 掌握 ECMAScript 6 中类、对象、生成器函数等相关语法；
3. 理解 ECMAScript 6 中的异步编程模型。

二、实验任务

1. 设计一个 HTML 页面，在页面中添加 JavaScript 代码，使其能够生成若干个随机数并从中找出和为特定值的 2 个数；
2. 设计一个 HTML 页面，在页面中添加 JavaScript 代码，使其能够每隔 1 秒显示一个斐波那契数，当数列长度达到 20 时结束。

三、实验步骤

1. 基于 ES6 编写一个函数，该函数能够随机生成 n 个元素的数组，数组的每个元素都是介于 min 与 max 之间的整数，其定义如下：

```
function genNumbers(n,min,max){  
    //补充代码  
}
```

2. 基于 ES6 编写一个函数，该函数能够从一个数组中的找出 2 个和为 sum 的元素，请将以下代码补充完整：

```
function findNumbers(nums,sum){  
    //补充代码  
}
```

3. 基于 ES6 定义一个类，该类能够代表一个数字卡片，请将以下代码补充完整：

```
class Card{  
    constructor(num,isAddend){  
        //补充代码  
    }  
    getHTML(){  
        //生成对应数字卡片的 HTML 代码  
    }  
}
```

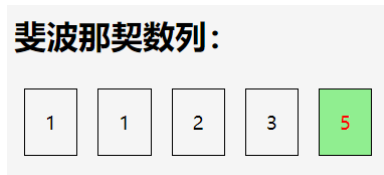
4. 利用上述定义的函数与类编写一个 HTML 页面，其最终效果如下：

寻找和为15的2个数：



页面上的数字随机生成，并突出显示和为特定值的 2 个数。

5. 创建一个斐波那契数列显示界面，其界面效果如下：



每秒显示一个数，以上分别是显示到第 5 个与第 10 个斐波那契数时的显示状态。

6. 以下是该页面中主要的 JavaScript 代码，请将其补充完整。

```
window.onload=displayDivs;

function* fabonaci(){
    let a = 1
    let b = 1
    for (let i=1; i<=10;i++){
        //补充代码
        let t = {id:i,num:b}
        yield t
    }
}

class Card{
    constructor(id,num){
        this.id = id
        this.num = num
    }
    getHTML(){
        return `<div id="num_${this.id}" class="numbers
changed">${this.num}</div>`
    }
}

async function displayDivs(){
    let fab = fabonaci()
    while(displayDiv(fab)){
        await sleep(1)
    }
}
```

```
function displayDiv(fab){  
  let {value,done} = fab.next()  
  if (!done){  
    //补充代码  
  }  
  return !done  
}  
  
function sleep(seconds){  
  return new Promise((resolve)=>{  
    setTimeout(function(){  
      resolve()  
    },seconds*1000)  
  })  
}
```

四、实验小结
(略)

实验二 HTML5 技术应用

一、实验目的

1. 熟悉 HTML5 中的新特性；
2. 掌握 HTML5 中 Canvas 的基本用法；
3. 理解 HTML5 中的 Web Worker 编程模型。

二、实验任务

1. 设计一个 HTML 页面，在页面中添加 JavaScript 代码，使其能够绘制螺旋图案；
2. 设计一个 HTML 页面，在页面中添加 JavaScript 代码，使其能够寻找指定范围内的孪生素数。

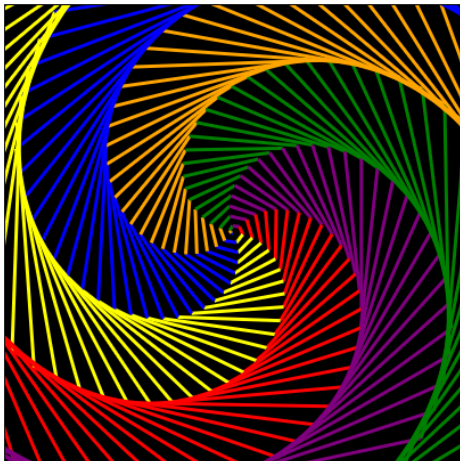
三、实验步骤

1. 设计一个 HTML 页面，在页面中添加 Canvas 标记：

```
<canvas id="drawingBoard" style="border:1px solid black; background: black" width=400 height=400></canvas>
```

2. 在页面上添加 JavaScript 代码，使其显示效果如下：

螺旋图案绘制



3. 创建一个名为 lab23.js 的 JavaScript 文件，并将其代码补全：

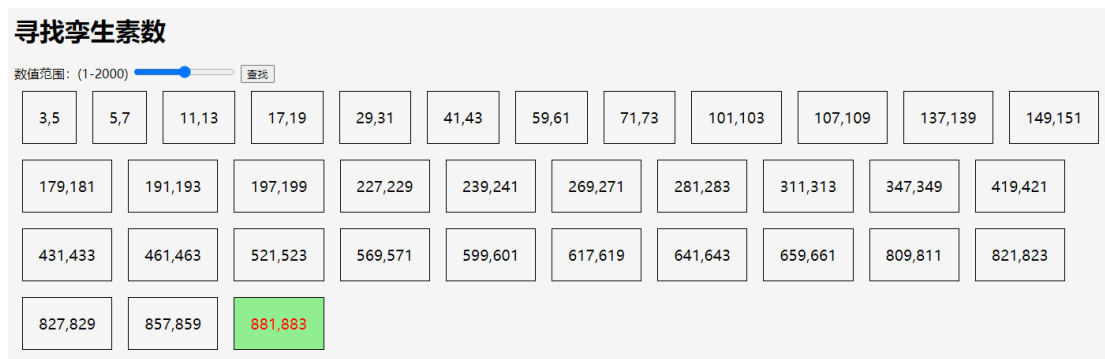
```
var primes = [2] //存放所有素数的数组
function findPrimeTwins(min,max){
    let i=1
    let n=min>3?min:3
    while(n<max){
        if (n==3 || isPrime(n)){
            //补全代码，每找到一对孪生素数，便向主线程发送消息
            primes.push(n)
        }
        n+=2
    }
}
```

```

}
function isPrime(n){
    let flag = true
    //补全代码，判断 n 是否是素数
    return flag
}
onmessage=(event)=>{
    let {min,max} = event.data
    findPrimeTwins(min,max)
}

```

4. 设计一个孪生素数查找页面，其显示效果如下：



5. 以下是该页面中的 Javascript 代码，请将其补全。

```

document.querySelector("input[type=button]").onclick=(()=>{
    let worker = new Worker("lab23.js")
    let max = document.getElementById("max").value
    worker.postMessage({min:1,max:max})
    document.getElementById("result").innerHTML = ""
    worker.onmessage=(event)=>{
        //补全代码，处理 Web Worker 发送过来的消息，显示孪生素数对
    }
}
class Card{
    constructor(id,num){
        this.id = id
        this.num = num
    }
    getHTML(){
        //补全代码，生成每张素数卡片所对应的 HTML 代码
    }
}

```

四、实验小结
(略)