JS手写题（50道）

JavaScript编程题集（50道）

基础语法题（1-15题）

1. 问题： 如何创建一个具有原型的对象？

答案：

代码块

```
let prototype = {
    greet: function() {
        console.log("Hello!");
    }
};
let obj = Object.create(prototype);
// 或者
function Person(name) {
    this.name = name;
}
Person.prototype.sayHello = function() {
    console.log(`Hello, I'm ${this.name}`);
};
let person = new Person("张三");
```

2. 问题： 写一个函数判断一个数字是否为偶数。

答案：

代码块

```
function isEven(num) {
    return num % 2 === 0;
}
// 或者箭头函数
const isEven = (num) => num % 2 === 0;
```

3. 问题： 如何创建一个包含5个元素的数组?

答案：

代码块

```
let arr = [1, 2, 3, 4, 5];
// 或者
let arr = new Array(5).fill(0);
// 或者
let arr = Array.from({length: 5}, (_, i) => i + 1);
```

4. 问题： 写一个函数计算两个数的和。

答案：

代码块

```
function add(a, b) {
    return a + b;
}
// 或者箭头函数
const add = (a, b) => a + b;
```

5. 问题： 如何获取字符串的长度?

答案：

代码块

```
let str = "Hello";
let length = str.length; // 5
```

6. 问题： 写一个for循环打印1到10的数字。

答案：

代码块

```
for (let i = 1; i <= 10; i++) {
    console.log(i);
```

```
}
```

7. 问题： 如何检查一个变量是否为undefined?

答案：

代码块

```
if (variable === undefined) {
    console.log("变量未定义");
}
// 或者
if (typeof variable === 'undefined') {
    console.log("变量未定义");
}
```

8. 问题： 写一个函数将华氏温度转换为摄氏温度。

答案：

代码块

```
function fahrenheitToCelsius(fahrenheit) {
    return (fahrenheit - 32) * 5/9;
}
```

9. 问题： 如何创建一个对象并添加属性?

答案：

代码块

```
let person = {
    name: "张三",
    age: 25,
    city: "北京"
};
// 或者
let person = {};
person.name = "张三";
person.age = 25;
```

10. 问题： 写一个函数检查字符串是否包含特定子字符串。

答案：

代码块

```
function containsSubstring(str, substring) {
    return str.includes(substring);
}
// 或者
function containsSubstring(str, substring) {
    return str.indexOf(substring) !== -1;
}
```

11. 问题： 如何将字符串转换为数字？

答案：

代码块

```
let str = "123";
let num = Number(str);
// 或者
let num = parseInt(str);
// 或者
let num = +str;
```

12. 问题： 写一个函数返回数组中的最大值。

答案：

代码块

```
function findMax(arr) {
    return Math.max(...arr);
}
// 或者
function findMax(arr) {
    let max = arr[0];
    for (let i = 1; i < arr.length; i++) {
        if (arr[i] > max) {
            max = arr[i];
```

```
            }
        }
        return max;
    }
```

13. 问题： 如何反转一个字符串？

答案：

代码块

```
function reverseString(str) {
    return str.split('').reverse().join('');
}
// 或者
function reverseString(str) {
    let reversed = '';
    for (let i = str.length - 1; i >= 0; i--) {
        reversed += str[i];
    }
    return reversed;
}
```

14. 问题： 写一个函数检查一个数是否为质数。

答案：

代码块

```
function isPrime(num) {
    if (num < 2) return false;
    for (let i = 2; i <= Math.sqrt(num); i++) {
        if (num % i === 0) return false;
    }
    return true;
}
```

15. 问题： 如何获取当前日期和时间？

答案：

```
let now = new Date();
console.log(now);
// 格式化输出
console.log(now.toLocaleDateString());
console.log(now.toLocaleTimeString());
```

数组操作题（16-25题）

16. 问题： 写一个函数移除数组中的重复元素。

答案：

代码块

```
function removeDuplicates(arr) {
    return [...new Set(arr)];
}
// 或者
function removeDuplicates(arr) {
    return arr.filter((item, index) => arr.indexOf(item) === index);
}
```

17. 问题： 如何将两个数组合并？

答案：

代码块

```
let arr1 = [1, 2, 3];
let arr2 = [4, 5, 6];
let merged = [...arr1, ...arr2];
// 或者
let merged = arr1.concat(arr2);
```

18. 问题： 写一个函数计算数组中所有数字的平均值。

答案：

代码块

```
function calculateAverage(arr) {
    let sum = arr.reduce((acc, num) => acc + num, 0);
    return sum / arr.length;
}
```

19. 问题： 如何对数组进行排序?

答案:

代码块

```
// 数字排序
let numbers = [3, 1, 4, 1, 5];
numbers.sort((a, b) => a - b); // 升序
numbers.sort((a, b) => b - a); // 降序

// 字符串排序
let strings = ['banana', 'apple', 'cherry'];
strings.sort(); // 字母顺序
```

20. 问题： 写一个函数查找数组中特定元素的索引。

答案:

代码块

```
function findIndex(arr, element) {
    return arr.indexOf(element);
}
// 或者使用findIndex方法
function findIndex(arr, condition) {
    return arr.findIndex(condition);
}
```

21. 问题： 如何过滤数组中的偶数?

答案:

代码块

```
function filterEvenNumbers(arr) {
    return arr.filter(num => num % 2 === 0);
```

```
}
```

22. 问题： 写一个函数将数组中的每个元素乘以2。

答案：

代码块

```
function multiplyByTwo(arr) {
    return arr.map(num => num * 2);
}
```

23. 问题： 如何检查数组中是否包含某个元素？

答案：

代码块

```
function containsElement(arr, element) {
    return arr.includes(element);
}
// 或者
function containsElement(arr, element) {
    return arr.indexOf(element) !== -1;
}
```

24. 问题： 写一个函数将二维数组扁平化。

答案：

代码块

```
function flattenArray(arr) {
    return arr.flat();
}
// 或者递归方法
function flattenArray(arr) {
    let result = [];
    for (let item of arr) {
        if (Array.isArray(item)) {
            result.push(...flattenArray(item));
        } else {
```

```
            result.push(item);
        }
    }
    return result;
}
```

25. 问题： 如何获取数组的最后一个元素？

答案：

代码块

```
function getLastElement(arr) {
    return arr[arr.length - 1];
}
// 或者
function getLastElement(arr) {
    return arr.slice(-1)[0];
}
```

对象操作题（26-35题）

26. 问题： 写一个函数复制对象的所有属性。

答案：

代码块

```
function copyObject(obj) {
    return {...obj};
}
// 或者
function copyObject(obj) {
    return Object.assign({}, obj);
}
// 深拷贝
function deepCopy(obj) {
    return JSON.parse(JSON.stringify(obj));
}
```

27. 问题： 如何获取对象的所有键？

答案：

```
代码块
1   function getObjectKeys(obj) {
2       return Object.keys(obj);
3   }
```

28. 问题： 写一个函数检查对象是否有特定属性。

答案：

```
代码块
1   function hasProperty(obj, prop) {
2       return obj.hasOwnProperty(prop);
3   }
4   // 或者
5   function hasProperty(obj, prop) {
6       return prop in obj;
7   }
```

29. 问题： 如何合并两个对象？

答案：

```
代码块
1   function mergeObjects(obj1, obj2) {
2       return {...obj1, ...obj2};
3   }
4   // 或者
5   function mergeObjects(obj1, obj2) {
6       return Object.assign({}, obj1, obj2);
7   }
```

30. 问题： 写一个函数计算对象中属性的数量。

答案：

```
代码块
1  function countProperties(obj) {
2      return Object.keys(obj).length;
3  }
```

31. 问题： 如何删除对象的某个属性？

答案：

```
代码块
1  function deleteProperty(obj, prop) {
2      delete obj[prop];
3      return obj;
4  }
5  // 或者创建新对象
6  function deleteProperty(obj, prop) {
7      let {[prop]: deleted, ...rest} = obj;
8      return rest;
9  }
```

32. 问题： 写一个函数将对象转换为数组。

答案：

```
代码块
1  function objectToArray(obj) {
2      return Object.entries(obj);
3  }
4  // 或者只要值
5  function objectToArray(obj) {
6      return Object.values(obj);
7  }
```

33. 问题： 如何检查一个值是否为对象？

答案：

```
代码块
1  function isObject(value) {
2      return typeof value === 'object' && value !== null &&
   !Array.isArray(value);
3  }
```

34. 问题： 写一个函数反转对象的键值对。

答案：

```
代码块
1  function reverseObject(obj) {
2      let reversed = {};
3      for (let key in obj) {
4          reversed[obj[key]] = key;
5      }
6      return reversed;
7  }
```

35. 问题： 如何创建一个具有原型的对象？

答案：

```
代码块
1  let prototype = {
2      greet: function() {
3          console.log("Hello!");
4      }
5  };
6  let obj = Object.create(prototype);
7  // 或者
8  function Person(name) {
9      this.name = name;
10 }
11 Person.prototype.sayHello = function() {
12     console.log(`Hello, I'm ${this.name}`);
13 };
14 let person = new Person("张三");
```

字符串处理题（36-45题）

36. 问题： 写一个函数统计字符串中每个字符的出现次数。

答案：

代码块

```
function countCharacters(str) {
    let count = {};
    for (let char of str) {
        count[char] = (count[char] || 0) + 1;
    }
    return count;
}
```

37. 问题： 如何将字符串的首字母大写？

答案：

代码块

```
function capitalizeFirst(str) {
    return str.charAt(0).toUpperCase() + str.slice(1);
}
```

38. 问题： 写一个函数检查字符串是否为回文。

答案：

代码块

```
function isPalindrome(str) {
    let cleaned = str.toLowerCase().replace(/[^a-z0-9]/g, '');
    return cleaned === cleaned.split('').reverse().join('');
}
```

39. 问题： 如何替换字符串中的所有空格为下划线？

答案：

```
代码块
1  function replaceSpaces(str) {
2      return str.replace(/ /g, '_');
3  }
4  // 或者
5  function replaceSpaces(str) {
6      return str.split(' ').join('_');
7  }
```

40. 问题： 写一个函数提取字符串中的所有数字。

答案：

```
代码块
1  function extractNumbers(str) {
2      return str.match(/\d+/g) || [];
3  }
```

41. 问题： 如何截取字符串的前n个字符？

答案：

```
代码块
1  function truncateString(str, n) {
2      return str.substring(0, n);
3  }
4  // 或者
5  function truncateString(str, n) {
6      return str.slice(0, n);
7  }
```

42. 问题： 写一个函数将字符串转换为驼峰命名法。

答案：

```
代码块
1  function toCamelCase(str) {
2      return str.replace(/[-_\s]+(.)?/g, (_, char) =>
```

```
            char ? char.toUpperCase() : ''
        );
    }
```

43. 问题： 如何计算字符串中单词的数量?

答案:

代码块

```
function countWords(str) {
    return str.trim().split(/\s+/).filter(word => word.length > 0).length;
}
```

44. 问题： 写一个函数移除字符串两端的空白字符。

答案:

代码块

```
function trimString(str) {
    return str.trim();
}
// 手动实现
function trimString(str) {
    return str.replace(/^\s+|\s+$/g, '');
}
```

45. 问题： 如何检查字符串是否以特定前缀开始?

答案:

代码块

```
function startsWith(str, prefix) {
    return str.startsWith(prefix);
}
// 手动实现
function startsWith(str, prefix) {
    return str.substring(0, prefix.length) === prefix;
}
```

高级应用题（46-50题）

46. 问题： 写一个函数实现防抖（debounce）。

答案：

代码块

```
function debounce(func, delay) {
    let timeoutId;
    return function(...args) {
        clearTimeout(timeoutId);
        timeoutId = setTimeout(() => func.apply(this, args), delay);
    };
}
```

47. 问题： 如何实现一个简单的Promise?

答案：

代码块

```
class SimplePromise {
    constructor(executor) {
        this.state = 'pending';
        this.value = undefined;
        this.onResolvedCallbacks = [];
        this.onRejectedCallbacks = [];

        const resolve = (value) => {
            if (this.state === 'pending') {
                this.state = 'resolved';
                this.value = value;
                this.onResolvedCallbacks.forEach(fn => fn());
            }
        };

        const reject = (reason) => {
            if (this.state === 'pending') {
                this.state = 'rejected';
                this.value = reason;
                this.onRejectedCallbacks.forEach(fn => fn());
```

```
            }
        };

        executor(resolve, reject);
    }

    then(onResolved, onRejected) {
        if (this.state === 'resolved') {
            onResolved(this.value);
        }
        if (this.state === 'rejected') {
            onRejected(this.value);
        }
        if (this.state === 'pending') {
            this.onResolvedCallbacks.push(() => onResolved(this.value));
            this.onRejectedCallbacks.push(() => onRejected(this.value));
        }
    }
}
```

48. 问题： 写一个函数实现深度克隆。

答案：

代码块

```
function deepClone(obj) {
    if (obj === null || typeof obj !== 'object') {
        return obj;
    }

    if (obj instanceof Date) {
        return new Date(obj.getTime());
    }

    if (obj instanceof Array) {
        return obj.map(item => deepClone(item));
    }

    if (typeof obj === 'object') {
        const cloned = {};
        for (let key in obj) {
            if (obj.hasOwnProperty(key)) {
                cloned[key] = deepClone(obj[key]);
            }
```

```
        }
        return cloned;
    }
}
```

49. 问题： 如何实现一个简单的发布订阅模式?

答案：

代码块

```
class EventEmitter {
    constructor() {
        this.events = {};
    }

    on(eventName, callback) {
        if (!this.events[eventName]) {
            this.events[eventName] = [];
        }
        this.events[eventName].push(callback);
    }

    emit(eventName, ...args) {
        if (this.events[eventName]) {
            this.events[eventName].forEach(callback => {
                callback(...args);
            });
        }
    }

    off(eventName, callback) {
        if (this.events[eventName]) {
            this.events[eventName] = this.events[eventName].filter(
                cb => cb !== callback
            );
        }
    }
}
```

50. 问题： 写一个函数实现柯里化（curry）。

答案：

代码块

```
function curry(fn) {
    return function curried(...args) {
        if (args.length >= fn.length) {
            return fn.apply(this, args);
        } else {
            return function(...args2) {
                return curried.apply(this, args.concat(args2));
            };
        }
    };
}

// 使用示例
function add(a, b, c) {
    return a + b + c;
}

const curriedAdd = curry(add);
console.log(curriedAdd(1)(2)(3)); // 6
console.log(curriedAdd(1, 2)(3)); // 6
```