20210619笔试

- 限时一个小时，四道题

JavaScript 相关

一、 找出数组中第K大和第M大的数字之和 **(15min)**

```
/**
 *
 * 示例:
 * let arr=[1,2,4,4,3,5],k=2,m=4
 * findTopSum(arr,k,m)
 * 第二大为4，第四大为2
 * 所以得 4*2+2*1=10
*/
var arr = [1, 2, 4, 4, 3, 5], k = 2, m = 4;
function findTopSum(arr, k, m) {
    var hash = {};
    //记录出现的频率
  for (const item of arr) {
        if (hash[item])
            hash[item] = hash[item] + 1
        else
            hash[item] = hash[item] = 1
    }
    //排序后去重
    arr = [...new Set(arr.sort((x, y) => { return x - y }))];
    //第K、M 大的值
    var paramK = arr[arr.length - k];
    var paramM = arr[arr.length - m];
    //第K大的值*频率 【下方判定当时笔试忘了写，所以当K、M超出数组范围会有bug】
    var sumK = paramK * hash[paramK] ? paramK * hash[paramK] : 0;
    var sumM = paramM * hash[paramM] ? paramM * hash[paramM] : 0;
    return sumK + sumM;
}
findTopSum(arr, k, m)
```

二、 清除对象中值为undefined或null的key并且返回新对象，不修改原对象。**(15min)**

- 如果不是使用reduce完成的，那么使用reduce如何完成?
  - 如果该规则允许自定义如何做?
  - 如果不允许有任何副作用，即不能修改reduce回调函数内的参数如何做?
  - 如果你可同时完成以上要求，那么你可以只使用一个函数

```
/** 题目 */
function clean(target){}
var target={a:undefined,b:null,c:false,d:'',e:0};
```

```
function cleanCust(/*自定义规则*/){ }
function cleanUseReduce(target){/*使用reduce完成*/ }
```

- 普通方法实现 /** 解答 clean(target); */

```
var target = { a: undefined, b: null, c: false, d: '', e: 0 };
var rule=['string','boolean'];

function clean(target) {
  let obj = {};
  for (const index in target) {
    var rule = typeof target[index] == 'undefined' || typeof target[index] ==
'object';
    if (!rule)
      obj[index] = target[index]
  }
  return obj;
}
```

- 带自定义规则方法实现 /** 解答 cleanCust(target); */

```
function cleanCust(target,rule) {
 let obj = {};
 //校验规则
 for (const i in target) {
   var filter= rule.some(function(item,index,array){
     return typeof target[i]==item;
   })
   if (!filter)
     obj[i] = target[i]
 }
 console.log(obj);
 return obj;
}
cleanCust(target,rule);
```

- reduce方法实现
  - 这个笔试写不出来，后来想了好久，写的非常丑陋，不知道怎么实现才好

```
/**
 * @params
 * target 存放原始对象
 * rules  存放验证规则，可自定义,不传默认过滤undefined、null
 * transArr 对象转为数组 [{x:'23'}, {a: undefined}..... {e: 0} ]
 * newobj 存放符合条件的对象
 */
var target = {x:'23', a: undefined, b: null, c: false, d: '', e: 0 };
```

```
function cleanUseReduce(target,rules){
 rules=rules?rules:['undefined','object']
 var transArr = [];
 var newobj={};
 for (let i in target) {
     let obj = {};
     obj[i] = target[i];
     transArr.push(obj)
 }
 transArr.reduce((currentValue, currentIndex, arr)=>{
   for (var index in currentIndex) {
     var falg=rules.includes(typeof currentIndex[index])
   }
   if(!falg)
      newobj[index]= target[index]
 },[])
 return newobj;
}
cleanUseReduce(target)
```

三、 防抖节流 **(15min)**

- 连续事件触发，每隔一段时间，只执行一次
- 连续事件触发，事件触发停止后，执行一次。

```
btn.addEventListener('click',throttle(submit,3000))
function submit(e){
  console.log('---');
}
function throttle(cbk,delay){
  // 设置初始值
  var begin=0;
  return(...args)=>{
      var cur=new Date().getTime();
      //每次点击的时间和上次的时间大于定义间隔则执行
      if(cur-begin>delay){
          cbk.apply(this,args)
          //更新初始值
          begin=cur;
      }
  }
}
```

```
btn = document.getElementById('btnId');
btn.addEventListener('click',debounce(submit,2000))
function submit(e){
    console.log('---');
}
function debounce(cbk,wait){
```

```
            let timer=null;
            return(...args)=>{
                if(timer) clearTimeout(timer)
                //避免函数自身污染外部arguments
                timer=setTimeout(() => {
                  timer=null ;
                  cbk.apply(this,args)
                }, wait);
            }
        }
```

四、 数据筛选 **(15min)**

- 一群学生，考语文数学两门，筛选出总分第一的同学
- 数据类型自定义，总分相同且第一，则筛选出并列

```
/**
 * @params
 * math, chinese 数学语文成功
 * Grades 最高分
 * pickName 最高分学生
 */
  var math = { xiaoMing: 66, xiaoHua: 66, xiaoHong: 95, xiaoLan: 99, xiaoBai: 68
};
  var chinese = { xiaoMing: 66, xiaoHua: 66, xiaoHong: 88, xiaoLan: 87, xiaoBai:
95 };
  function pick(M, C) {
    var Grades = 0;
    var pickName = '';
    for (const key in M) {
      M[key] = M[key] + C[key];
      if (M[key] > Grades)
        //后边总分比前大，则重设最高总分
        [Grades, pickName] = [M[key], key]
      else if (M[key] == Grades)
        //与最高分相同则一同记录
        pickName += key;
    }
    console.log(`最高分是${Grades}--${pickName}`);
  }
  pick(math, chinese)
```