## 每日一题 - 数据格式转化问题

### 信息卡片

* 时间：2019-08-05
* tag：格式 对象

### 题目描述

已知数据格式：

const userToSkill = {  
 robert: ["programming", "design", "reactjs"],  
 kimia: ["java", "backend", "services"],  
 patrick: ["reactjs"],  
 chris: ["reactjs", "programming"]  
 };

转化成

const skillToUser = {  
 programming: ["robert", "chris"],  
 reactjs: ["patrick", "robert", "chris"],  
 java: ["kimia"],  
 backend: ["kimia"],  
 services: ["kimia"],  
 design: ["robert"]  
 };

扩展：

完成函数：

function setOfUsersWithSimillarSkills(userToSkill) {  
  
}

期望输出： ["robert", "chris"]

### 参考代码

方案一：

function transform(userToSkill) {  
 const skillToUser = {};  
 for (const k in userToSkill) {  
 userToSkill[k].forEach(skill => {  
 if (skillToUser[skill] === void 0) {  
 skillToUser[skill] = [k];  
 } else {  
 skillToUser[skill].push(k);  
 }  
 });  
 }  
  
 return skillToUser;  
}

方案二：

const transform = userToSkill =>  
 Object.entries(userToSkill).reduce((skillToUser, [user, skills]) =>  
 skills.reduce((skillToUser, skill) =>({  
 ...skillToUser,  
 [skill]: skillToUser[skill] ? skillToUser[skill].concat(user) : [skill]  
 }), skillToUser), {})

扩展：

function backtrack(list, tempList, nums, start) {  
 list.push(intersect(...tempList));  
 for(let i = start; i < nums.length; i++) {  
 tempList.push(nums[i]);  
 backtrack(list, tempList, nums, i + 1);  
 tempList.pop();  
 }  
}  
/\*\*  
\* @param {number[]} nums  
\* @return {number[][]}  
\*/  
function subsets(nums) {  
 const list = [];  
 backtrack(list, [], nums, 0);  
 return list;  
};  
  
function intersect(arr1, ...rest) {  
 const set = new Set();  
 if (rest.length === 0) return set;  
 const s1 = new Set(arr1);  
 const restSet = rest.map(a => new Set(a));  
 for (const x of s1) {  
 if (restSet.every(s => s.has(x))) {  
 set.add(x);  
 }  
 }  
 return set;  
}  
function maxByLength(list) {  
 let max = 0;  
 let maxIndex = 0;  
  
 for(let i = 0; i < list.length; i++) {  
 if (list.length > max) {  
 max = list.length;  
 maxIndex = i;  
 }  
 }  
 return list[maxIndex];  
}  
  
function setOfUsersWithSimillarSkills(userToSkill) {  
 const mapper = transform(userToSkill);  
 const sets = subsets(Object.values(mapper).filter(q => q.length > 1));  
 const list = sets.map(set => Array.from(set)).filter(q => q.length);  
 return maxByLength(list)  
}