// 1.对下面数组按年龄和姓名排序，要求不使用 Array sort 方法

const data = [{age:2,name:'mapple'},{age:7,name:'qapple'},{age:3,name:'fapple'},{age:1,name:'tapple1'},{age:7,name:'aapple1'},];

// 2.将扁平数组转换为树状结构

// 部门 -> 子部门 扁平数组测试数据

const arr = [

{id: 1, name: '部门1', pid: 0}, // 根部门

{id: 2, name: '部门2', pid: 1}, // 根部门 的子部门

{id: 3, name: '部门3', pid: 1}, // 根部门 的子部门

{id: 4, name: '部门4', pid: 3}, // 部门3 的子部门

{id: 5, name: '部门5', pid: 4}, // 部门4 的子部门

]

/\*

输出结果

[

{

"id": 1,

"name": "部门1",

"pid": 0,

"children": [

{

"id": 2,

"name": "部门2",

"pid": 1,

"children": []

},

{

"id": 3,

"name": "部门3",

"pid": 1,

"children": [

// 结果 ,,,

]

}

]

}

]

\*/

// 3.按顺序写出输出结果，并说明原理

const promise = new Promise((resolve, reject) => {

console.log(1);

setTimeout(() => {

console.log("timerStart");

resolve("success");

console.log("timerEnd");

}, 0);

console.log(2);

});

promise.then((res) => {

console.log(res);

});

console.log(4);

/\*\*

4.二选一

使用 React 高阶组件实现条件渲染

使用 Redux 实现一个待办应用。包含待办名称、是否完成、新增待办、完成代办

\*/