

//这是问题的图解，，，，，构建三个hash表来存储一些信息根据题目

//首先构造从起点抵达当前节点的距离hash表

var length=new Map();

length.set('a',0);length.set('e',9);length.set('c',1);length.set('b',Number.MAX\_VALUE);length.set('d',Number.MAX\_VALUE);length.set('over',Number.MAX\_VALUE)

//然后构造每个节点的父节点初始值

var path=new Map();

path.set('a','');path.set('e','a');path.set('c','a');path.set('b','c');path.set('d','b');path.set('over','e')

//构建节点间关系hash表

var Node=new Map([['a',new Map([['e',9],['c',1]])],['e',new Map([['over',1]])],['c',new Map([['b',1],['over',7]])],['b',new Map([['d',1]])],['d',new Map([['e',1]])],['over','']])

//创建判断是否处理过的表

var handle=[];

//找到下一个节点函数，，，这里很简单，，就是遍历length表，，，找到到起点开销最小的节点

function findnextNode(length){

var cost=Number.MAX\_VALUE;

var costnode=null;

for(let [key, value] of length) {

if(handle.indexOf(key)==-1&&value<cost){

cost=value;

costnode=key

}

}

return costnode

}

//这里是主要代码

node=findnextNode(length)//首先找到第一个

while(node){

cost=length.get(node)//获取当前节点的开销

var neighbours=Node.get(node)//获取当前节点的邻居

for (var [key, value] of neighbours) {// 循环遍历当前节点的 邻居，

var newcost=cost+value //临时存储当前节点的开销加上当前节点到邻居的开销的和

if(newcost<length.get(key)){//如果和小于当前存储的开销

length.set(key,newcost);//更新开销

path.set(key,node)//更新其父节点

}

}

handle.push(node) //当前节点标记为已处理

node=findnextNode(length)//寻找下一个未处理的节点

}

//最后打印

console.log("最短距离为"+length.get('over'));

var over='over'

var resultpath=''

while(over){

resultpath=over+resultpath

over=path.get(over)

}

console.log("最短路径为"+resultpath)