let commonKeysResult = {}

function checkObject\_common(object) {

if( typeof(object) !== 'object' || object === null || Array.isArray(object)) throw 'Kindly provide valid object as input'

//if (Object.keys(object).length === 0) throw 'Object is blank'

return true

}

function objectNest(key,obj) {

let temp\_obj = {}

if(typeof(obj) === 'object'){

for(let j in obj){

//temp\_obj = {}

//temp\_obj = objectNest(j,obj[j])

objectNest(j,obj[j])

}

}/\* else{

temp\_obj = {}

temp\_obj[key] = obj

return temp\_obj

} \*/

if(!(key in commonKeysResult)){

commonKeysResult[key] = new Array()

commonKeysResult[key].push(obj)

}else{

commonKeysResult[key].push(obj)

}

return true

}

function checkIP(value) {

if (value === null || typeof(value) === 'undefined') throw 'Invalid value'

}

function checkIPKey(value) {

//console.log(! isNan(value) )

console.log(typeof(value))

if (value === null || typeof(value) === 'undefined') throw 'Invalid value'

}

function commonKeys(obj1,obj2) {

checkObject\_common(obj1)

checkObject\_common(obj2)

commonKeysResult = {}

for(let i in obj1){

checkIP(i)

checkIPKey(obj1[i])

if(typeof(obj1[i]) === 'object'){

objectNest(i,obj1[i])

}else{

if(!(i in commonKeysResult)){

commonKeysResult[i] = new Array()

commonKeysResult[i].push(obj1[i])

}else{

commonKeysResult[i].push(obj1[i])

}

}

}

for(let i in obj2){

checkIP(i)

checkIPKey(obj2[i])

if(typeof(obj2[i]) === 'object'){

objectNest(i,obj2[i])

}else{

if(!(i in commonKeysResult)){

commonKeysResult[i] = new Array()

commonKeysResult[i].push(obj2[i])

}else{

commonKeysResult[i].push(obj2[i])

}

}

}

let commonKeyObjects = {}

for(k in commonKeysResult){

if(Object.keys(commonKeysResult[k].length >= 2 )){

if (commonKeysResult[k][0] === commonKeysResult[k][1]){

commonKeyObjects[k] = commonKeysResult[k][0]

}

}

}

return commonKeyObjects

}

const first = {a: 2, b: 4};

const second = {a: 5, b: 4};

const third = {a: 2, b: {x: 7}};

const fourth = {a: 3, b: {x: 7, y: 10}};

//console.log(commonKeys(third,fourth))

console.log(commonKeys({a:{b:{c:d}}},{a:{b:{c:d}}}))

--------------------------- Finally YEAH BHENCHODDD-----------

let commonKeysResult = {}

function checkObject\_common(object) {

if( typeof(object) !== 'object' || object === null || Array.isArray(object)) throw 'Kindly provide valid object as input'

//if (Object.keys(object).length === 0) throw 'Object is blank'

return true

}

function objectNest(key,obj) {

let temp\_obj = {}

if(typeof(obj) === 'object'){

for(let j in obj){

temp\_obj = {}

temp\_obj[key] = objectNest(j,obj[j])

//objectNest(j,obj[j])

return temp\_obj

}

}else{

temp\_obj = {}

temp\_obj[key] = obj

return temp\_obj

}

if(!(key in commonKeysResult)){

commonKeysResult[key] = new Array()

commonKeysResult[key].push(obj)

}else{

commonKeysResult[key].push(obj)

}

return true

}

function checkIP(value) {

if (value === null || typeof(value) === 'undefined') throw 'Invalid value'

}

function checkIPKey(value) {

//console.log(! isNan(value) )

//console.log(typeof(value))

if (value === null || typeof(value) === 'undefined') throw 'Invalid value'

}

/\* function printResultObjects(object1,object2){

for(j in object1){

for(h in object2){

if( j === h && objet1[j] === object2[h]){

commonKeyObjects[j] = commonKeysResult[k][0][j]

}

}

}

} \*/

function commonKeys(obj1,obj2) {

checkObject\_common(obj1)

checkObject\_common(obj2)

commonKeysResult = {}

let temp\_objs = {}

for(let i in obj1){

checkIP(i)

checkIPKey(obj1[i])

if(typeof(obj1[i]) === 'object'){

temp\_objs = {}

temp\_objs = objectNest(i,obj1[i])

for(let l in temp\_objs){

if(!(i in commonKeysResult)){

commonKeysResult[i] = new Array()

commonKeysResult[i].push(temp\_objs[i])

}else{

commonKeysResult[i].push(temp\_objs[i])

}

}

}else{

if(!(i in commonKeysResult)){

commonKeysResult[i] = new Array()

commonKeysResult[i].push(obj1[i])

}else{

commonKeysResult[i].push(obj1[i])

}

}

}

for(let i in obj2){

checkIP(i)

checkIPKey(obj2[i])

if(typeof(obj2[i]) === 'object'){

temp\_objs = {}

temp\_objs = objectNest(i,obj2[i])

for(let l in temp\_objs){

if(!(i in commonKeysResult)){

commonKeysResult[i] = new Array()

commonKeysResult[i].push(temp\_objs[i])

}else{

commonKeysResult[i].push(temp\_objs[i])

}

}

}else{

if(!(i in commonKeysResult)){

commonKeysResult[i] = new Array()

commonKeysResult[i].push(obj2[i])

}else{

commonKeysResult[i].push(obj2[i])

}

}

}

let commonKeyObjects = {}

// console.log(commonKeysResult)

for(k in commonKeysResult){

if(Object.keys(commonKeysResult[k].length >= 2 )){

if (typeof(commonKeysResult[k][0]) === 'object' && typeof(commonKeysResult[k][1]) === 'object'){

//this will go in recursion

//let printResult = printResultObjects(commonKeysResult[k][0],commonKeysResult[k][1])

for(j in commonKeysResult[k][0]){

for(h in commonKeysResult[k][1]){

if( j === h && commonKeysResult[k][0][j] === commonKeysResult[k][1][h]){

commonKeyObjects[k] = commonKeysResult[k][0]

}

}

}

//this will go in recursion

}

if (commonKeysResult[k][0] === commonKeysResult[k][1]){

commonKeyObjects[k] = commonKeysResult[k][0]

}

}

}

return commonKeyObjects

}

const first = {a: 2, b: 4};

const second = {a: 5, b: 4};

const third = {a: 2, b: {x: 7, y:10}};

const fourth = {a: 3, b: {x: 7, y: 10}};

console.log(commonKeys(third,fourth))

console.log(commonKeys(third,second))

console.log(commonKeys(third,first))

console.log(commonKeys(fourth,first))

//console.log(commonKeys({a:{b:{c:d}}},{a:{b:{c:d}}}))